AFIT/DS/ENP/96-01

CHARACTERISTIC SPATIAL QUADRATURES FOR DISCRETE ORDINATES NEUTRAL PARTICLE TRANSPORT ON ARBITRARY TETRAHEDRAL MESHES

DISSERTATION

Charles R. Brennan, Captain, USAF

AFIT/DS/ENP/96-01

Approved for public release; distribution unlimited

AFIT/DS/ENP/96-01

CHARACTERISTIC SPATIAL QUADRATURES FOR DISCRETE ORDINATES NEUTRAL PARTICLE TRANSPORT ON ARBITRARY TETRAHEDRAL MESHES

DISSERTATION

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

Charles R. Brennan, B.S., M.S.

Captain, USAF

June 1996

AFIT/DS/ENP/96-01

CHARACTERISTIC SPATIAL QUADRATURES FOR DISCRETE ORDINATES NEUTRAL PARTICLE TRANSPORT ON ARBITRARY TETRAHEDRAL MESHES

Charles R. Brennan, B.S., M.S.
Captain, USAF

Approved:

Kirk A. Mathews — 30 May 96
Kirk A. Mathews
Associate Professor of Nuclear Engineering
Research Advisor

Wm. F. Bailey — 23 May 96
William F. Bailey
Associate Professor of Physics

D L Coulliette — 23 May 96
David L. Coulliette
Assistant Professor of Applied Mathematics

Todd A. Wareing — 13 May 96
Todd A. Wareing
Radiation Transport Team
Los Alamos National Laboratory

Wm P. Baker — 25 May 96
William P. Baker
Associate Professor of Applied Mathematics
Dean's Representative

Accepted:

Robert A. Calico
Dean, Graduate School of Engineering

Acknowledgments

I am eternally grateful to the many people who assisted me during my research. I owe a large debt to Captains Glenn Sjonden, Dennis Miller and Bryan Minor for building the foundation upon which my work rests. I wish to thank Dr. David Weeks for all of his patience when I bombarded him with questions in the early stages of the project. I'm also extremely grateful to Dr. Wally Walters, an original committee member who passed away during my research, for his encouragement and help in defining a research topic. I am indebted to the members of my committee, Dr. William Bailey, Lt. Col. Dave Coulliette, and Dr. Todd Wareing, for their advice, support, and encouragement. I'm especially grateful to my research advisor, Dr. Kirk Mathews, for bravely agreeing to take me on as his student, for laying much of the groundwork leading up to this project, for his advice and assistance in defining the research topic, for displaying an infinite amount of patience, and for guiding me (successfully) through to the end. Most importantly, I thank my wife Fran, and my children, Tiffany and Meaghan. None of this would have been possible without their patient understanding and unending support.

Table of Contents

List of Figures

List of Tables

Abstract

Characteristic spatial quadratures for discrete ordinates calculations on meshes of arbitrary tetrahedra are derived and tested, including the step (SC), linear (LC), and exponential (EC) characteristic quadratures and variants that assume constant distributions on cell faces. Tetrahedral meshes accurately model curved surfaces with few cells. A split cell approach subdivides tetrahedra along the streaming direction, reducing the transport to one dimension. Assumed forms of the cell source and entering flux distributions have sufficient parameters to match the zeroth and first spatial moments. These parameters are determined by analytically inverting a linear system (LC), or by numerical inversion using Newton's method (EC). Efficient algorithms for the two- and three-dimensional rootsolves are derived. The constant face methods proved unacceptable in empirical testing. Both LC and EC exhibited third order convergence. LC provided accurate results on cells with optical thickness on the order of one mean free path while EC was accurate with fewer, thicker cells. LC can produce negative fluxes; EC is strictly positive. Although more costly per cell, EC is robust and can be more efficient than LC or SC by using coarse meshes.

CHARACTERISTIC SPATIAL QUADRATURES FOR DISCRETE ORDINATES NEUTRAL PARTICLE TRANSPORT ON ARBITRARY TETRAHEDRAL MESHES

I. Introduction

Existing codes for performing three-dimensional discrete ordinates calculations are generally limited to rectangular parallelpiped cells and low-order spatial quadratures. The spatial quadratures currently employed in these codes are simple to implement and well understood, but are also very inaccurate unless a highly refined spatial mesh is employed. Rectangular parallelpiped cells are especially limiting due to their inability to adequately resolve curved boundaries or inclined surfaces without excessive refinement. In certain situations, the level of mesh refinement required to adequately resolve boundaries may be greater than that required to ensure accuracy of the transport solution on the individual cells. Non-rectangular cells defined in terms of curvilinear coordinates can be used to alleviate some of the difficulties, but many of the higher order solution techniques are not amenable to derivation in curvilinear coordinates. This research addresses these limitations by deriving and implementing highly accurate characteristic spatial quadratures for meshes of arbitrary tetrahedra.

1.1. Background

The fundamental equation describing the neutral particle transport process is the linear Boltzmann transport equation (BTE) (Lewis and Miller, 1993: 1) . The BTE describes the balance of neutral particles in a seven-dimensional phase space: three spatial dimensions, two angular dimensions, energy, and time (Lewis and Miller, 1993: 24-42). The BTE relates the time rate of change of the neutral particle density at a point r, moving with energy E, in direction Ω, at time t, to the difference between sources and losses:

$$\left[\frac{1}{v}\frac{\partial}{\partial t} + \hat{\Omega}\cdot\vec{\nabla} + \sigma(\vec{r},E,t)\right]\psi(\vec{r},E,\hat{\Omega},t) = S(\vec{r},E,\hat{\Omega},t)\ , \qquad (1)$$

where the angular flux density, ψ, is the flux of particles at point r, with energy E, moving in direction Ω, at time t, the bracketed term accounts for particle losses via streaming and collisions, and the right hand side accounts for all sources of particles. The equation is examined in detail in section 2.1. Analytic solutions to the BTE exist only for highly idealized cases. In general, solutions are obtained using approximate, numerical techniques. In certain cases, the diffusion equation, which can be derived from the BTE using several simplifying assumptions about the angular dependence, can be used to obtain approximate solutions. Methods for solving the fully angular dependent BTE fall into two broad categories: Monte Carlo and deterministic.

1.1.1 Diffusion Approximation

Accounting for the angular dependence of the transported particles introduces considerable complexity and represents a large part of the computational effort in most solution methods. For many applications this added complexity is unnecessary and reasonable results can be achieved by assuming that the flux is only weakly dependent on angle, i.e. that the flux is linearly anisotropic (Duderstadt, 1976:133-140). Under this assumption, the BTE can be used as a starting point in the derivation of the diffusion equation:

$$\frac{1}{v}\frac{\partial \phi}{\partial t} - \vec{\nabla}\cdot D(\vec{r},E)\vec{\nabla}\phi(\vec{r},E,t) + \sigma_t(\vec{r},E)\phi(\vec{r},E,t) = \int_0^\infty dE'\,\sigma_s(E'\to E)\phi(\vec{r},E',t) + S(\vec{r},E,t)\ . \tag{2}$$

Where D, the diffusion coefficient, determines the rate at which the diffusion occurs, and the scalar flux, ϕ, is defined as

$$\phi(\vec{r},E,t) = \int d\hat{\Omega}\,\psi(\vec{r},\hat{\Omega},E,t)\ . \tag{3}$$

Since the assumption of weak angular dependence works well in the cores of thermal nuclear reactors, the diffusion approximation is widely used in the nuclear industry. However, the application of diffusion theory requires that the migration process be dominated by scatter, and that the process be far removed from any sources or sinks, i.e. large gradients which introduce an angular dependence. However, for some very

important applications, such as near control rods in thermal reactors, blankets in fast reactors, or deep penetration shielding type problems, the diffusion approximation is inadequate, and transport theory must be used (Duderstadt, 1979:138).

1.1.2. Monte Carlo Methods

Monte Carlo is a technique by which individual particle's paths are simulated. These histories are accumulated and statistics are formed to estimate desired processes. One advantage of Monte Carlo is its ability to handle complex three-dimensional geometries. Another advantage of Monte Carlo is the continuous treatment of space and angle variables which eliminates the discretization errors associated with deterministic methods. Monte Carlo errors take the form of stochastic uncertainties (Lewis and Miller, 1993: 296). One disadvantage of the technique is that the accuracy of the estimators is dependent on statistical variance. Reducing the variance may require a large number of samples, which is computationally expensive. Some variance reduction techniques are available and improve the accuracy of estimators while reducing the computational expense. However, use of these techniques is as much art as science (Lewis and Miller, 1993: 297-298).

1.1.3. Deterministic Methods

Unlike Monte Carlo, which can be understood without reference to the BTE, deterministic methods solve the BTE in one form or another by using various techniques to treat the angular and spatial integrations. The most widely used method for solving the BTE is discrete ordinates (Lewis and Miller, 1993: 116). In discrete ordinates, the angular

variable is discretized, and the BTE is solved for a discrete set of directions, or ordinates. The angular integrations then reduce to weighted sums. The spatial variable is also discretized, and treated using an appropriate spatial quadrature. The discrete ordinates method was first utilized to perform transport calculations in stellar atmospheres (Chandrasekhar: 1960). The method was adapted for use in the neutron transport community by Carlson and others working at Los Alamos National Laboratory (Carlson, 2386: 1958; Carlson and Lathrop: 1968)

The basic computational unit in a discrete ordinates calculation is the spatial cell, and as such, computational cost and storage requirements are directly proportional to the number of cells in the calculation. Initial discrete ordinates codes utilized the diamond difference (DD) spatial quadrature which uses a finite difference approximation to the derivative term in the BTE along with several auxiliary assumptions (Lewis and Miller, 1993: 128-131). The diamond difference method is easy to implement and computationally inexpensive, but can be very inaccurate on cells with optical thickness greater than about a tenth of a mean free path (mfp). On thicker cells (in slab geometry) and in two or three spatial dimensions, the method may generate unphysical negative fluxes. In two-dimensional rectangular geometry, unphysical oscillations in the DD generated solutions have been observed (Petrovic and Haghighat: 1995). The shortcomings of diamond difference prompted efforts to develop more robust schemes leading to the development of the linear discontinuous (Lewis and Miller, 1993: 134-135), exponential (Barbucci and Di Pasquantonio: 1977), linear nodal (Walters and O'Dell,

1981) and linear characteristic (Larsen and Alcouffe, 1981) methods. The linear nodal and linear characteristic methods provide accurate solutions on cells with optical thickness on the order of 1 mfp (generally 2-4 mfp), but can still produce negative fluxes on thicker cells.

The desire to have good coarse mesh accuracy and strict positivity prompted the development of the nonlinear characteristic methods: step adaptive (SA), linear adaptive (LA) and most recently, exponential characteristic (EC). Each of the methods solve the BTE on a spatial cell by integrating along a characteristic. Assumed forms of the source distribution are used having parameters that are chosen to match the spatial moments of the distribution on the spatial cell. The SA, LA, and EC methods use increasingly smoother source distribution approximations. The SA method assumes a piecewise constant source distribution (Mathews: 1991; Mathews and Minor: 1993), and the LA method assumes a piecewise linear cell source distribution (Mathews and Minor: 1993). The exponential characteristic method (Sjonden: 1992; Mathews, Sjonden and Minor: 1994; Walters and Wareing: 1994; Minor: 1993; Mathews and Brennan: 1995) uses an exponential source distribution which can be thought of as an extension of the SA and LA distributions to a C^{∞} function. All of the nonlinear methods have the desirable property of being strictly positive. The EC method, in addition to being strictly positive exhibits excellent accuracy on optically thick cells (optical thickness > 10 mfp).

Due to the relatively high computational cost per spatial cell of the EC method, efforts have begun to develop related methods that use exponential source and flux

representations, but are less computationally intensive. These methods include the exponential discontinuous (ED) method (Walters and Wareing: 1996) and the nonlinear corner balance method (NLCB) (Castrianni and Adams: 1995). Both of these methods are relatively new and as of the present time, their performance relative to the EC method has not been firmly established.

Central to the development of improved solution methods has been the application of the methods to unstructured grids. Monte Carlo methods are readily adapted to general three-dimensional problems, making them the method of choice for three-dimensional transport calculations. While there are many situations for which Monte Carlo is the appropriate choice, there are problems for which discrete ordinates could be more efficient, particularly when spatial distributions are required. The current generation of three-dimensional discrete ordinates codes use diamond difference or linear nodal spatial differencing (Alcouffe: 1995; Rhoades and Azmy: 1995) and meshes of rectangular parallelpipeds. Rectangular parallelpiped meshes are especially limiting for several reasons. First, many small cells are required to adequately resolve curved boundaries or inclined surfaces. Further, if the mesh needs to be refined to provide higher resolution in a small region of the problem, it must be refined everywhere. Decreasing the cell size by a factor of n in each direction results in an n^3 increase in the total number of cells, and a corresponding n^3 increase in computation cost and storage.

General geometry methods for performing efficient neutral particle transport in multiple dimensions are beginning to be developed. There have been several recent efforts

to develop methods for solving the BTE on unstructured meshes of cells in two-dimensions. Corner-balance schemes (Adams: 1993; Castrianni and Adams: 1995) solve the BTE on an unstructured grid of polygons. The step and linear characteristic methods have been extended to 2D triangular (Miller: 1993) and arbitrary polygonal (Groves and Pevey: 1995) meshes, and the exponential characteristic method has also been applied to unstructured meshes of triangles (Mathews and Brennan: 1995).

The goal of this research is to derive and implement the characteristic methods on meshes of arbitrary (unstructured) tetrahedra. This will address the need for a general geometry, three-dimensional discrete ordinates transport code that may provide a more computationally efficient alternative to Monte Carlo and the existing three-dimensional discrete ordinates codes.

The characteristic methods allow for accurate solutions on large, or optically thick, cells (cell through which the streaming length is many mean free paths) by integrating the transport equation along streaming, or characteristic, lines of particle flow (characteristic lines of the BTE). One drawback of the method is the fact that the characteristic integrations must be analytically evaluated to derive the methods. In curvilinear coordinates, the characteristics are not straight lines, and the resulting integrations are intractable in those coordinate systems. However, curvilinear coordinates are useful only for problems having the required symmetries. For example, cylindrical coordinates work well for a cylinder, or a set of concentric cylinders, but are useless for two cylinders with distinct axes.

Using meshes of arbitrarily oriented tetrahedra addresses many of the previously mentioned difficulties associated with discrete ordinates solution methods. First, they are defined in terms of a Cartesian coordinate system, so the characteristics are straight lines, and the resulting integrations can be analytically evaluated. Second, curved surfaces that required the use of a large number of parallelpiped cells to model can be reasonably approximated using a much smaller collection of arbitrary tetrahedra. Finally, global mesh refinements are not necessary. Mesh refinements in localized regions of the problem can be accomplished with minimum impact on the size of the cells in the rest of the mesh.

1.2. Purpose

The purpose of this research is to derive, implement, and evaluate the step, linear, and exponential characteristic methods for meshes of arbitrary tetrahedra. In addition, variants of the linear and exponential characteristic methods using constant boundary flux representations will also be examined. Dr. W. F. (Wally) Walters suggested investigating the constant face methods to determine if the higher order source distribution approximation would increase the efficiency of the step characteristic method (Walters, 1994).

Several of the required tools were brought forward from previous efforts. The algorithm used to test the methods is a three-dimensional adaptation of Miller's (1993) arbitrary triangle algorithm for the LC method on meshes of arbitrary triangles. In deriving the EC method for 2D rectangular meshes, Minor (1993) developed routines to numerically compute exponential moment functions of all orders with up to three

arguments. For the tetrahedral mesh methods, routines to compute four-argument moment functions were developed, and all of the multi-dimensional functions routines were rewritten to increase their efficiency.

Specifically, the implementation of the quadratures requires:

a. Extending the triangle discrete ordinates algorithm (Miller, 1993) to three dimensions.

b. Developing definitions of, appropriate coordinate systems for, and optimal means of passing spatial moments on tetrahedral faces.

c. Developing methods for splitting arbitrary tetrahedra along the particle streaming direction and generating the sub-tetrahedra and necessary transformations between the parent and sub-tetrahedra coordinate systems.

d. deriving equations for computing the cell and exiting flux moments for each of the methods in the appropriate coordinate systems.

e. Developing efficient algorithms for determining coefficients of the entering flux and source distributions using knowledge of the zeroth and first spatial moments.

f. Developing efficient routines to evaluate multi-dimensional exponential moment functions.

g. Developing a method to generate the required tetrahedral meshes and create appropriate data structures for input to the algorithm.

1.3. Scope

Although with straightforward modification, the methods derived here can be applied to a wide range of problems, the derivations include only enough complexity to demonstrate the effectiveness of the spatial quadratures on meshes of arbitrary tetrahedra. First, only the time-independent, or steady state case is considered. Time dependent problems are beyond the scope of this research.

As the spatial quadratures essentially invert the streaming-collision operator on the left hand side of the BTE, the complete details of the source term on the right hand side of the equation need not be considered to demonstrate the methods. The derivations assume that all scatter is isotropic. Although the addition of anisotropic scatter to the method is straightforward (Lewis and Miller, 1993:61-77), its inclusion is not necessary to demonstrate correct operation of the methods and was therefore omitted. Similarly, only non-multiplying, or no fission, cases were considered. Sub-critical multiplication simply requires the inclusion of an additional term on the right hand side of the equation, and critical and super-critical systems require both a slightly altered form of the BTE and the modified source term, but neither requires any modification to the spatial quadratures (Lewis and Miller, 1993:90-103).

No type of acceleration technique (Lewis and Miller, 1993: 145-150) was examined for accelerating the convergence of the methods for optically thick problems with large scattering to total cross section ratios. There are many problems for which the methods as derived are entirely applicable. At this point it is not clear if the any of the methods need to be modified in any way to accommodate an accelerator, but the ability to handle problems with cross sections approaching the diffusion limit is beyond the scope of the current effort. It is anticipated, however, that a linearization acceleration technique would be applicable to the EC method developed here (Wareing, Walters, and Morel, 1995).

1.4. Organization

The remainder of this document consists of five chapters and four appendices. Chapter II discusses the discrete ordinates method, identifies the role of the spatial quadrature, and introduces the general form of the characteristic spatial quadratures. Chapter III develops the characteristic methods for arbitrary tetrahedral meshes, including development of coordinate systems, the splitting of arbitrary tetrahedra to produce special case tetrahedra, the special case moment equations for each of the methods derived, and the derivation and uses of the balance equations. Chapter IV presents the solution algorithm, the details of the iterative solution procedure, and concludes with a pseudocode outline of the entire code. Chapter V presents the results of numerical tests of the code on several test problems and reports on the basic operating characteristics of the code including convergence rates, sensitivity to mesh variations, speed, and accuracy. The final chapter presents my conclusions and recommendations for further efforts.

There are five appendices that provide a more detailed discussion of procedures used in the algorithm. Appendix A details the numerical algorithm used to evaluate the exponential moments functions. Appendix B deals with the derivation and operation of the two- and three-dimensional root-solving routines. Appendix C provides further details of the tetrahedral splitting routines. Appendix D contains a discussion of the use of the balance equations to compute cell flux moments. Appendix E discusses mesh generation using the IDEAS (Lawry, 1991) code.

II. Derivation of the Characteristic Spatial Quadratures

2.1. The Boltzmann Transport Equation

The BTE, introduced in Section 1.1, describes the balance of neutral particles in a seven-dimensional phase space. The BTE relates the time rate of change of the neutral particle density at a point r, moving with energy E, in direction Ω, at time t, to the difference between sources and losses:

$$\frac{1}{v}\frac{\partial}{\partial t}\psi(\vec{r},E,\hat{\Omega},t)+\left[\hat{\Omega}\cdot\vec{\nabla}+\sigma(\vec{r},E,t)\right]\psi(\vec{r},E,\hat{\Omega},t)=S(\vec{r},E,\hat{\Omega},t)\,, \qquad (4)$$

where the angular flux density, ψ, is the flux of particles at point r, with energy E, moving in direction Ω, at time t. The bracketed term on the left-hand side of the equation is termed the streaming-collision operator. The streaming-collision operator consists of the particle-streaming-loss and the collision-loss terms. The total interaction cross section, σ, represents the probability at point r, per unit path length of particle travel, that a particle having energy E will undergo a collision. The total cross section term in the streaming collision operator accounts for losses caused by both scatter and absorption, $\sigma = \sigma_s + \sigma_a$. Particles with a given energy and direction, removed by scatter, become a source of particles in other energies and directions. Particles lost by absorption can also result in sources to other energies and directions via secondary emission from fission, (n, 2n), (n, γ), (n, α) and other similar reactions.

The right hand side of the equation contains the sum of all sources of particles, and can be expanded into

$$S(\vec{r},E,\hat{\Omega},t) = q_s(\vec{r},E,\hat{\Omega},t) + q_f(\vec{r},E,\hat{\Omega},t) + q_{ext}(\vec{r},E,\hat{\Omega},t) \ , \qquad (5)$$

which contains contributions from scatter, q_s, fission, q_f, and other (extrinsic) sources, q_{ext}.

The scattering source term, q_s, accounts for the production of particles with energy E moving in direction Ω as a result of scattering collisions,

$$q_s(\vec{r},E,\hat{\Omega},t) = \int dE' \int d\hat{\Omega}' \, \sigma(\vec{r},E' \to E, \hat{\Omega}' \to \hat{\Omega},t) \, \psi(\vec{r},E',\hat{\Omega}',t) \ . \qquad (6)$$

Equation (6) states that the source of particles with energy E, moving in direction Ω, at point r, and time t, is simply the sum of all particles scattered into energy E and direction Ω, from other energies E' and directions Ω'.

The fission term, q_f, accounts for the production of particles (specifically neutrons) with a given energy and direction due to fission,

$$q_f(\vec{r},E,\hat{\Omega},t) = \chi(E) \int dE' \int d\hat{\Omega}' \, \nu\sigma_f(\vec{r},E',t) \, \psi(\vec{r},E',\hat{\Omega}',t) \ . \qquad (7)$$

The term $\chi(E)$ accounts for the fact that the energy distribution of fission neutrons is independent of the energy of the particle that caused the fission. Since number of neutrons produced is the desired quantity, it is conventional to use $\nu\sigma_f$, the product of the fission cross section, σ_f, and the average number of neutron emitted per fission, ν. Note that the fission cross section contains no reference to the incident or exiting neutron

directions. This is because fission reactions produce neutrons with an isotropic angular distribution, and the probability of fission is independent of the direction of motion of the incident neutron.

The extrinsic source term, q_{ext}, accounts for particles produced by means other than scattering or fission. This may include material sources, such as particles produced as a result of spontaneous fission or decay, cosmic rays, etc.

Although the BTE is solved for ψ, the angular flux itself is not a physically useful quantity. The desired products are usually particle currents or reaction rates. A more useful quantity for computing reaction rates is the scalar flux, ϕ, defined by

$$\phi(\vec{r},E,t) = \int d\hat{\Omega}\, \psi(\vec{r},E,\hat{\Omega},t) \; . \tag{8}$$

Whereas the angular flux distribution, $\psi\, d\vec{r}\, dE\, d\hat{\Omega}$, can be thought of as the total path length per unit time traveled by particles in the phase space volume $(d\vec{r}\, dE\, d\hat{\Omega})$ moving in a given direction, Ω, the scalar flux distribution, $\phi\, d\vec{r}\, dE$, is just the total path length rate of particles in the phase space volume traveling $(d\vec{r}\, dE)$ in all directions. Because reaction cross sections are almost always independent of the incident particle direction, reaction rates are the product of the reaction cross section and the scalar flux.

Particle leakage rates are governed by currents. Again, one is generally interested in the total number of particles penetrating a shield or leaking through a crack, not in the specific angular distribution of the penetrating particles. The current vector is defined as

$$\vec{J}(\vec{r},E,t) = \int d\Omega\, \hat{\Omega}\, \psi(\vec{r},E,\hat{\Omega},t) \,. \tag{9}$$

The net number of particles per unit time per unit energy per unit surface area crossing a surface traveling in a direction with a component in the $\hat{n}$ (the outward surface normal) direction is then just

$$J_n(\vec{r},E,t) = \int d\Omega\, \hat{n}\cdot\hat{\Omega}\; \psi(\vec{r},E,\hat{\Omega},t) = \hat{n}\cdot\vec{J}(\vec{r},E,t) \,. \tag{10}$$

As is obvious from the equations, computing the angular flux correctly is essential to computing the scalar flux and current correctly.

2.2. Discrete Ordinates Formulation

The time-independent form of the BTE in Cartesian coordinates is

$$\left[\mu\frac{\partial}{\partial x} + \eta\frac{\partial}{\partial y} + \xi\frac{\partial}{\partial z} + \sigma(x,y,z,E)\right]\psi(x,y,z,E,\hat{\Omega}) = S(x,y,z,E,\hat{\Omega}) \,. \tag{11}$$

where $\mu = \hat{e}_x\cdot\hat{\Omega}$, $\eta = \hat{e}_y\cdot\hat{\Omega}$, and $\xi = \hat{e}_z\cdot\hat{\Omega}$ are the direction cosines of $\hat{\Omega}$. Defining the differential solid angle, $d\Omega$ as

$$d\Omega = \frac{d\omega}{2\pi}\frac{d\theta\sin\theta}{2} = \frac{d\omega}{2\pi}\frac{d\xi}{2} \,, \tag{12}$$

gives the normalization,

$$\int d\Omega = \int_0^{2\pi} \frac{d\omega}{2\pi} \int_{-1}^{1} \frac{d\xi}{2} = 1 \tag{13}$$

(Lewis and Miller, 1993, 28). Assuming a non-multiplying system and isotropic scatter, and using equations (8) and (13), the source term reduces to

$$S(x,y,z,E) = \int dE' \, \sigma_s(x,y,z,E' \to E) \, \phi(x,y,z,E') + q_{ext}(x,y,z,E) \; . \tag{14}$$

Rather than solving over a continuous angular variable, the discrete ordinates method solves the BTE for a discrete set of directions, or ordinates. The discretized (in angle) form of Equation (11) for a single direction, $\hat{\Omega}_n$, is

$$\left[\mu_n \frac{\partial}{\partial x} + \eta_n \frac{\partial}{\partial y} + \xi_n \frac{\partial}{\partial z} + \sigma(x,y,z,E) \right] \psi_n(x,y,z,E) = S(x,y,z,E) \; . \tag{15}$$

The angular integration used to compute the scalar flux and current is now approximated by a weighted sum of the form

$$\phi(x,y,z,E) = \sum_{n=1}^{N} w_n \, \psi_n(x,y,z,E) \; . \tag{16}$$

The set of direction cosines, (μ_n, η_n, ξ_n) and weights, w_n, is termed the angular quadrature set. Quadrature sets are chosen both to assure accuracy of the numerical integrations and to provide required symmetries for reflective boundary conditions. The

set of angular quadratures that are invariant under 90 degree rotations about any axis are called the level symmetric quadratures. If the symmetry requirement can be relaxed, other quadratures, tailored to the specifics of the problem, can be derived (Lewis and Miller, 1993: 162-165; Abu-Shumays, 1977).

The directions associated with a given quadrature set also determine the directions in which the particles are allowed to travel. Allowing particle flow in only a discrete set of directions can lead to unphysical behaviors of the flux, called ray effects (Lewis and Miller, 1993: 194-203), as described in section 2.7.

The energy dependence is also handled by discretization. In this case, the energy spectrum is divided into a set of discrete bins, or energy groups. The discretization produces a set of equations (15) for each energy group. These equations are coupled through the scattering term because as particles lose (or gain) energy through scatter, they are transported from group to group. Discretizing equation (15) in energy yields

$$\left[\mu_n \frac{\partial}{\partial x} + \eta_n \frac{\partial}{\partial y} + \xi_n \frac{\partial}{\partial z} + \sigma_g(x,y,z)\right]\psi_{n,g}(x,y,z) = S_g(x,y,z)\ . \tag{17}$$

The group fluxes, $\psi_{n,g}$ and $\phi_{n,g}$, are given by

$$\psi_{n,g}(x,y,z) = \int_{E_g}^{E_{g-1}} dE'\psi_n(x,y,z,E')\ , \tag{18}$$

$$\text{and}, \qquad \phi_{n,g}(x,y,z) = \sum_{n=1}^{N} w_n \psi_{n,g}(x,y,z) \tag{19}$$

where the limits of integration are the lower and upper bounds of energy group g. The group source term is given by

$$S_g(x,y,z) = \sum_{g'=1}^{g-1} \sigma_{g'\to g}(x,y,z)\,\phi_{g'}(x,y,z) + \sigma_{g\to g}(x,y,z)\,\phi_g(x,y,z) + q_{ext,g}(x,y,z)\,, \tag{20}$$

where the group scattering cross section, $\sigma_{g'\to g}$, is the probability that a particle in energy group g' undergoing a collision will emerge in energy group g. The group subscripts increase for decreasing energy: group 1 is the highest energy group; group G is the lowest. Equation (20) only allows for down-scatter, i.e. particles may not gain energy in collisions. For many problems, including photon transport and fast reactors, this assumption is strictly valid (Lewis and Miller, 1993, 79). For thermal reactors, there is only appreciable up scatter for neutrons with thermal energies. In this case equation (20) can be used if the energy range in which up scatter occurs is contained in a single energy group (Duderstadt, 1976, 291-295).

Spatial discretization is accomplished by dividing the domain into a set of spatial cells. The cross sections are assumed constant within each spatial cell. There is then an equation (17) for each spatial cell for each angle and energy group. Once discretized, the

solution is obtained using a spatial quadrature to invert the streaming-collision operator in equation (17).

2.2.1 Boundary Conditions

To solve the BTE, the flux distribution entering the solution volume, V, across the bounding surface, Γ, must be known. Specifically, if $\hat{n}$ has an outward orientation with respect to Γ, then $\psi_{n,g}(\vec{r})$ for $\hat{n}\cdot\hat{\Omega}_n < 0$ and $\vec{r}\in\Gamma$ are required as boundary data. For a known incident flux, $\tilde{\Psi}_{n,g}(\vec{r})$, the boundary condition is

$$\psi_{n,g}(\vec{r}) = \tilde{\Psi}_{n,g}(\vec{r}), \quad \hat{\Omega}_n\cdot\hat{n} < 0, \quad \vec{r}\in\Gamma. \tag{21}$$

A vacuum boundary condition results when $\tilde{\Psi}_{n,g}(\vec{r}) = 0$.

The surface source and vacuum boundary conditions are explicit. Implicit boundary conditions, can be used to take advantage of symmetries. An albedo boundary condition is given by

$$\psi_{n,g}(\vec{r}) = \alpha_g\,\psi_{n',g}(\vec{r}), \quad \hat{\Omega}\cdot\hat{n} < 0, \ \vec{r}\in\Gamma, \tag{22}$$

where α_g is the albedo and $\hat{\Omega}$ is the angle resulting from specular reflection of particles traveling in the $\hat{\Omega}_{n'}$ direction. The special case where $\alpha = 1$ is termed a symmetry boundary. Other types of implicit boundaries include the white boundary, in which all particles are reflected back from a surface with an isotropic (Lambertian) distribution, and the periodic boundary condition, where the distribution entering through one boundary is set equal to the distribution leaving through the corresponding boundary in a periodic

lattice (Lewis and Miller, 1993: 25-26). The current implementation of the code supports both types of explicit boundary conditions and the albedo boundary condition.

2.3. Discrete Ordinates Solution

The coupled set of equations (17), (19), and (20), together with the appropriate boundary conditions, are solved using an iterative procedure termed the iteration on the scattering source. The procedure consists of an outer iteration through each of the energy groups, and in inner iteration for each energy group that consists of a loop through each of the angles in the quadrature, and a loop through each of the cells in the spatial mesh. The outer iteration is the iteration on the between group scattering source in equation (19). The inner iteration is the iteration on the within group scattering source.

The inner iteration consists of a "walk" through the spatial mesh for each angle until the within group scattering source for the current energy group is converged. The angle determines the direction of particle flow, and hence the direction of walk through the spatial mesh. Equation (17) is solved for each cell in the spatial mesh using a spatial quadrature, defined in detail in Section 2.4. Input information to the spatial quadrature consists of spatial moments of the fluxes on the entering faces of the cell (determined by the current streaming direction) and the source within the cell. Initially, the cell source is determined using an initial value for the group scalar fluxes in equation (19) (usually zero) to compute the group source for each cell in the spatial mesh. Given the source and entering flux information, the cell interior and exiting fluxes are computed using the spatial quadrature. The exiting fluxes for the current cell then become the entering fluxes for the

neighboring downstream cells. The procedure continues in the direction of particle flow until each of the cells in the spatial mesh have been computed. The current angle contribution to the scalar flux is now known and added to a running total using equation (16). A new angle is then chosen and the procedure repeated until all the angles in the quadrature set have been used. At this point, each term in the sum of equation (16) has been accumulated. The current group scalar fluxes for each cell are compared to those from the previous iteration. If the relative change in values is larger than some predetermined tolerance, the source term for each cell is updated using the new group scalar flux in equation (19), the current iteration scalar flux value is saved and used for comparison at the end of the next iteration, and the iteration procedure repeated. If the relative change in values is less than some predetermined tolerance, the method is said to have converged. The current group scalar flux is then used in equation (19) to update the down-scatter source for the lower-energy groups. This completes an inner iteration. This procedure is then repeated for each of the energy groups, completing the outer iteration.

An outline of the discrete ordinates iterative procedure follows:

Determine initial values for the group scalar fluxes

For each energy group

Do:

Update scattering source with previous iteration scalar fluxes

For each angle:

For each spatial cell:

Compute cell and exiting angular flux data

Pass cell exiting flux data to downstream cells as input information

Update scalar flux sum with current cell angular flux

Next Spatial Cell

Next angle

Check for convergence by comparing previous iteration scalar flux to current value

Loop until converged

Update lower energy group scattering sources with current group scalar flux

Next Energy Group

Convergence of the method can be explained by examining the physical interpretation of the iteration on the scattering source. Initially, the scattering source is zero (or some initial guess), and all particles in the problem are either present initially, arise from internal sources, or enter through the problem boundaries. Since none of these particles have scattered, the flux computed in the first iteration is termed the first-flight, or uncollided, flux. For the second iteration, the source consists of particles arising from extrinsic sources together with a contribution from particles that have been scattered once. The third iteration source contains contributions from particles that have been scattered at most twice, etc. If the physics of the problem is such that the average particle undergoes only a few scatters before either being absorbed or leaking out of one of the boundaries,

then successive contributions to the scattering source from particles that have undergone one more scatter should ultimately decrease to the point where the updated angular fluxes are not significantly different from the previous iteration fluxes. At this point the method is said to have converged.

2.4. Role of the Spatial Quadrature

The spatial quadrature numerically inverts the streaming-collision operator of equation (17) on a computational cell. The inputs to the spatial quadrature are the spatial moments of the scattering source distribution within the cell and the flux distribution on the entering face(s) of the cell. The spatial quadratures derived here use assumed analytic forms of the input distributions with parameters chosen to match either the zeroth or both the zeroth and first moments of the distributions. Using these parameters, the quadrature computes the moments of the flux distribution within the cell volume and on the exiting face(s).

The spatial moments are defined as weighted integrals over either the cell face or cell volume. The zeroth moment is simply the average, and the weighting function is one. The first moment describes how the flux or source is distributed with respect to one of the coordinate axes, and is weighted accordingly. Moments of the entering flux distribution are defined as

$$\begin{pmatrix} \psi_A^{in} \\ \psi_{s_1}^{in} \\ \psi_{s_2}^{in} \end{pmatrix} = \frac{1}{A_{face}} \int\int dA_{face} \begin{pmatrix} 1 \\ s_1 \\ s_2 \end{pmatrix} \psi^{in}[\vec{r}(s_1, s_2)] , \tag{23}$$

where ψ^{in} (s_1, s_2) is the assumed form of the entering flux distribution, and

$$\vec{r}(s_1,s_2) = \vec{e}_o + s_1\hat{e}_1 + s_2\hat{e}_2 , \tag{24}$$

where $\hat{e}_o$ is a convenient point on the face, such as a vertex, and $\hat{e}_1$ and $\hat{e}_2$ are orthogonal unit vectors that span the plane of the face. The moments of the source distribution are defined as

$$\begin{pmatrix} S_A \\ S_{x1} \\ S_{x2} \\ S_{x3} \end{pmatrix} = \frac{1}{V_{cell}} \int\int\int dV_{cell} \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} S(x_1,x_2,x_3) , \tag{25}$$

where $S(x_1, x_2, x_3)$ is the assumed form of the source distribution, and x_1, x_2, and x_3 parameterize the cell volume.

The actual values of the entering flux moments are known, either from the boundary conditions or passed from a upstream cell. The values of the source distribution moments are also known, either from an initial guess or computed using previous iteration results. Equations (24) and (25) are evaluated analytically giving systems of equations in these known moments having the parameters of the assumed distributions as unknowns. The parameters are then determined by inverting the system, either by direct analytical or numerical inversion for linear systems, or by a multi-dimensional root solve for non-linear systems.

The spatial quadrature inverts equation (17), giving an expression for the cell flux distribution in terms of the source and entering flux parameters. This procedure is described in detail in the following section. Once an expression for the flux in the cell is known, it can be used to determine the moments of the cell and exiting flux distributions. The moments of the cell flux distribution are defined as

$$\begin{pmatrix} \psi_A^{cell} \\ \psi_{x1}^{cell} \\ \psi_{x2}^{cell} \\ \psi_{x3}^{cell} \end{pmatrix} = \frac{1}{V_{cell}} \iiint dV_{cell} \begin{pmatrix} 1 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} \psi^{cell}(x_1, x_2, x_3), \tag{26}$$

and the moments of the exiting flux distribution are defined as

$$\begin{pmatrix} \psi_A^{out} \\ \psi_{s1}^{out} \\ \psi_{s2}^{out} \end{pmatrix} = \frac{1}{A_{face}} \iint dA_{face} \begin{pmatrix} 1 \\ s_1 \\ s_2 \end{pmatrix} \psi^{cell}[x_1(s_1, s_2), x_2(s_1, s_2), x_3(s_1, s_2)] \tag{27}$$

where $\psi^{cell}(x_1, x_2, x_3)$ is the cell flux distribution given in terms of the entering flux and scattering source parameters, and s_1 and s_2 are defined for the exiting face. The cell angular flux moments are used in equation (19) to generate the scalar flux moments which are then used to update the scattering source distribution for the next iteration:

$$\begin{pmatrix} S_{A,g} \\ S_{x,g} \\ S_{y,g} \\ S_{z,g} \end{pmatrix}^{k} = \sum_{g'=g+1}^{G} \sigma_{s,g'\to g} \begin{pmatrix} \phi_{A,g'} \\ \phi_{x,g'} \\ \phi_{y,g'} \\ \phi_{z,g'} \end{pmatrix} + \sigma_{s,g\to g} \begin{pmatrix} \phi_{A,g} \\ \phi_{x,g} \\ \phi_{y,g} \\ \phi_{z,g} \end{pmatrix}^{k-1} + \begin{pmatrix} S_A^{ext} \\ S_x^{ext} \\ S_y^{ext} \\ S_z^{ext} \end{pmatrix} \qquad (28)$$

where G is the total number of energy groups, the subscript g indicates the group number, and the superscript k indicates the iteration number (Note that equation (28) assumes down-scatter only). The exiting flux moments become the entering flux moments for downstream cells.

Using the details of the spatial quadrature, the discrete ordinates iterative algorithm is

Initialize scalar fluxes

For each energy group:

Do:

Update source moments using previous iteration scalar fluxes

Determine the scattering source distribution parameters for each spatial cell using known source moments

For each angle:

For each spatial cell:

Determine the parameters of the entering face flux distributions from the known moments

Compute cell and exiting flux moments using the entering and source distribution parameters

Pass cell exiting flux moments to downstream cells as input information

Accumulate current angle contribution to scalar flux sum

Next spatial cell

Next Angle

Check for within group convergence by checking relative error between current and previous iteration scalar fluxes

Loop Until Converged

Update lower energy group scattering sources using current group scalar fluxes

Next energy group

2.5. Moments of the Boltzmann Transport Equation

The moments integrals can also be applied to the discretized BTE to produce conservation or balance equations. The conservation equations describe the balance of particles over the spatial cell. Existing spatial quadratures utilize either only the zeroth or the zeroth and first moments. Methods that use only the zeroth moment satisfy the zeroth moment balance equation, methods that use both zeroth and first moments satisfy both the zeroth and first moment balance equation. In general, methods that utilize first moments exhibit higher orders of convergence and are more accurate than methods that use only the zeroth moment. However, the cost of increased accuracy and faster convergence is increased computational effort and storage cost for each cell (Lewis and Miller, 1993:

135). Nevertheless, the more complicated methods can be more efficient by producing the same accuracy with fewer cells.

2.6. Characteristic Solutions to the Boltzmann Transport Equation

The angular and energy dependence of the BTE are handled by iterating on a discretized form of the equation. The spatial dependence is treated by use of the spatial quadrature. Characteristic spatial quadratures solve the discretized form of the BTE, Equation (17), by integrating along a characteristic, or particle streaming direction. The general form of the characteristic solution for a single ordinate has the form

$$\psi(\vec{r},\hat{\Omega}_n) = \psi(\vec{r}_0,\hat{\Omega}_n)e^{-\sigma_t s} + \int_0^s ds' e^{-\sigma_t(s-s')} S(s') , \tag{29}$$

where s is the position along the characteristic line starting at $\vec{r}_0$ traveling in the $\hat{\Omega}_n$ direction, and $\vec{r} = \vec{r}_0 + s\hat{\Omega}_n$ (Alcouffe, R.E. and E.W. Larsen, 1981:3-6). The different characteristic methods are distinguished by the assumed form of the entering flux distribution, $\psi(\vec{r}_0,\hat{\Omega}_n)$, and source distribution, S(s).

2.6.1. Step Characteristic

The step characteristic (SC) method assumes that the entering flux and source distributions are constant,

$$\begin{aligned} S(x,y,z) &= S_A \\ \psi^{in}(s_1,s_2) &= \psi_A^{in} , \end{aligned} \tag{30}$$

where s_1 and s_2 are parameters defining points on the entering face. The SC method is computationally inexpensive compared to higher order methods. The method satisfies only the zeroth moment balance equation. The method has been demonstrated to be first-order convergent in slab geometry (Larsen and Miller, 1980: 78) and in two-dimensional rectangular and triangular geometries (Minor, 1993: 89-95; Miller, Mathews and Brennan, 1996).

The representation of the entering flux and source distributions as constants can lead to inaccuracies caused by numerical diffusion. The inaccuracies and low order of convergence resulting from the assumption of constant entering flux and source suggest the use of a higher order approximation to the distributions.

2.6.2. Linear Characteristic

The linear characteristic (LC) method models the source and entering flux as linear distributions within the cell and on the entering face:

$$\begin{aligned} S(x,y,z) &= a + bx + cy + dz, \\ \psi^{in}(s_1,s_2) &= \alpha + \beta_1 s_1 + \beta_2 s_2 . \end{aligned} \tag{31}$$

The LC method has been extensively evaluated and has been shown to perform well on a wide range of problems (Larsen and Miller, 1980: 106-107; Mathews, Sjonden, and Minor, 1994:33-36; Miller, 1993: 70-113; Minor, 1993: 83-139) . The method conserves both zeroth and first moments, and has been shown to be fourth order convergent in slab geometry (Larsen and Miller, 1980: 80) and third order convergent in two dimensional

rectangular and triangular geometries. (Larsen and Alcouffe, 1981; Miller, Mathews and Brennan, 1996).

The linear representation of the source in the spatial cell can lead to problems for optically thick cells, i.e. cells that are more than a few mean-free-paths thick. If the source distribution is such that the linear approximation is negative on the exiting face of the cell, an unphysical negative flux can result (Minor, 1993:9-12). Should this occur, it can be remedied in one of three ways. First, the cell size can be reduced, thereby reducing the optical thickness of all cells. This approach increases the number of cells required to model the geometry, and increases the computational cost. A second solution is to invoke a negative flux fix up. There are several procedures available for fixing up negative fluxes, but each destroys first moment conservation, degrading the accuracy of the solution (Larsen and Alcouffe, 1981: 101-106). At third approach is to abandon the linear approximation and choose a distribution that is strictly positive and more realistically models the actual distribution.

2.6.3. Exponential Characteristic

The exponential characteristic (EC) method models both the entering flux and source distributions as exponentials,

$$S(x,y,z) = ae^{bx+cy+dz}$$
$$\text{and} \quad \psi^{in}(s_1,s_2) = \alpha e^{\beta_1 s_1 + \beta_2 s_2}. \tag{32}$$

In optically thick cells without embedded sources, the exponential distribution provides a reasonable approximation to the actual distributions. Further, the exponential distribution has the added advantage of being strictly positive. The method conserves both zeroth and first moments, and displays fourth order convergence in slab geometry (Mathews, Sjonden, and Minor, 1994: 33-37) and third order in convergence in two dimensional rectangular (Minor, 1993: 89-95) and triangular (Mathews and Brennan, 1995) geometry. The method has been demonstrated to perform exceptionally well in deep shielding type problems, where the distributions are strongly attenuated across the domain of the problem (Mathews, Sjonden, and Minor, 1994; Walters and Wareing, 1994; Minor and Mathews, 1995; and Mathews and Brennan, 1995). When initially introduced, the method was much less computationally efficient per spatial cell than existing quadratures (approximately 20 times the cost of LC), but its accuracy on coarse meshes allowed it to obtain highly accurate results using a relatively small number of cells resulting in less overall computational effort and greatly reduced storage requirements (Mathews, Sjonden, and Minor, 1993; Minor, 1993; Mathews and Brennan, 1996). However, recent improvements in the implementation and special function algorithms have reduced the cost to about twice that of the LC method per spatial cell on triangular meshes (Mathews and Brennan, 1996).

2.6.4. LC and EC Variants - Constant Face Methods

Variants of the first moment conserving methods (LC, EC) can be derived in which the source distribution is expanded in a linear or exponential function but the face flux is

assumed constant. These methods are also derived, implemented, and tested as part of this project.

2.7. Discretization Errors

Discretization of the space and angle variables invariably introduces errors. In two or more dimensions, systematic errors caused by angular discretization can occur. In problems with small scattering-to-total cross section ratios, the neutron distribution may contain peaks that coincide with the streaming directions of the particles that are inconsistent with the physical realities of the problem. These non-physical oscillations are known as ray effects. Ray effects can be reduced by several means including increasing the order of the quadrature or reverting to a product quadrature (Lewis and Miller, 1993: 195-203; Abu-Shumays, 1977).

Spatial discretization can also contribute to a non-physical redistribution of particles known as numerical diffusion. Numerical diffusion is caused by smearing the input flux distribution over the entire face of the cell or of the source distribution over the interior of the cell, either of which could result in an erroneous redistribution of particles throughout the problem. In some cases, the numerical diffusion and ray effects can, to an extent, mask each other. Particles incorrectly bunched along a streaming direction due to ray effects can to some extent be fortuitously redistributed almost correctly by the effects of the numerical diffusion, nevertheless, the solution accuracy is degraded by both effects. Numerical diffusion can be mitigated by carrying first moments of the distribution along

the cell faces. This helps to maintain the proper distribution of the angular flux as it is passed between cells (Mathews, 1983: VI-5 - VI-8).

III. Derivation of the Characteristic Methods for Arbitrary Tetrahedral Meshes

3.1. Application of the Characteristic Methods to Arbitrary Tetrahedral Meshes

Trying to solve the transport equation on a mesh of arbitrary tetrahedra, where everything is defined in terms of a single, global coordinate system introduces prohibitive complexity. In essence, under such a scheme every integration used to compute cell moments must be considered as a special case. It is possible to choose a local coordinate system for each tetrahedron that standardizes the form of the source and flux distribution moments. This, however, complicates the characteristic solution because the characteristics are arbitrary in the local coordinate system, and the characteristic integration has to be treated as a special case for each tetrahedron. Miller (1993) solved these difficulties in two dimensions by developing a procedure whereby arbitrary triangles were split along the characteristic direction. The tetrahedral mesh algorithm presented here is a direct extension of Miller's technique to three dimensions.

Consider a tetrahedron oriented in space in such a way that the particle streaming direction is coincident with one of the edges, as shown in Figure 1. The local u,v,w coordinate system is oriented such that the u-axis is coincident with the streaming direction and the base of the tetrahedron lies in the uv plane. We term this special case tetrahedron a case 1 tetrahedron, referring to the fact that there is only one inflow face and

one outflow face. For a case 1 tetrahedron in the given coordinate system, the one-group form of equation (17) reduces to the one-dimensional form,

$$\left[\frac{\partial}{\partial u} + \sigma\right]\psi(u,v,w) = S(u,v,w)\ , \tag{33}$$

where $\hat{\Omega}_n = \hat{e}_u$, $\mu_n = \hat{\Omega}_n \cdot \hat{e}_u = 1$, $\eta_n = \hat{\Omega}_n \cdot \hat{e}_v = 0$ and, $\xi = \hat{\Omega}_n \cdot \hat{e}_w = 0$ and we assume that the cross sections are constant within the cell. Because the characteristic direction is parallel to the u-axis, the characteristic equation, equation (29), becomes

$$\psi(u,v,w) = \psi_{in}\left[u_{in}(v,w),v,w\right] e^{-\sigma[u-u_{in}(v,w)]} + \int_0^u du' e^{-\sigma(u-u')} S(u',v,w)\ . \tag{34}$$

Substitution of the assumed form of the entering flux and source distribution into equation (34) gives a functional form for the cell flux distribution in terms of the entering flux and source distribution coefficients. This can then be used to compute the flux distribution moments in the cell volume and on the cell outflow face.

3.2. Splitting of Arbitrary Tetrahedra into Sub-Tetrahedra

The algorithm developed here uses a splitting routine that divides arbitrarily-oriented tetrahedra along the streaming direction such that a number of sub-tetrahedra are produced, each of which has exactly one input and one output face. Each such subcell is a case 1 tetrahedron because the edge at the intersection of the remaining two faces is oriented along the streaming direction. Using this method, a local coordinate system can

be erected for each sub-tetrahedron identical to that shown in Figure 1. The characteristic equation is then identical for all sub-tetrahedra and only a single, consistent set of analytic equations for the angular flux moments are required; no special cases need be considered. This simplifies both the derivation of the analytic form of the moments and the computation of the numerical values of the moments.

The possible results of tetrahedral splitting are shown in Figure 2. Each possibility is labeled by the resulting number of case 1 tetrahedra resulting from the split. In reality, there are two general cases, case 4 and case 3, and two degenerate cases, case 2 and case 1. If the streaming direction is incident on the tetrahedron such that there are two inflow and two outflow faces, the tetrahedron is case 4. The case 4 tetrahedron is split along a line parallel to the streaming direction that enters one edge and exits the opposite edge resulting in four case 1 tetrahedra.

When three sub-tetrahedra result, the tetrahedron is either case 3 or case -3. A case 3 tetrahedron is characterized by having three input faces and one output face. A case -3 tetrahedron has one input face and three output faces. A case 3 tetrahedron is split along a line parallel to the streaming direction that enters through an apex and exits through the opposite (outflow) face. A case -3 tetrahedron is split along a line parallel to the streaming direction that enters through a face and exits through the apex opposite the inflow face.

A case 2 tetrahedron has two entering faces and one exiting face. Likewise, a case -2 tetrahedron has one entering face and two exiting faces. In either case there is one face

parallel to the streaming direction. A case 2 tetrahedron is split along a line parallel to the streaming direction that enters through an apex and exits the face opposite the apex. A case -2 tetrahedron is split along a line that enters through a face and exits through the opposite apex. Either case can be seen to be a degenerate case 3 tetrahedron with the entering or exiting point on the face located on an edge.

Appendix C contains a more detailed discussion of each case and provides the details needed to implement the splitting procedure.

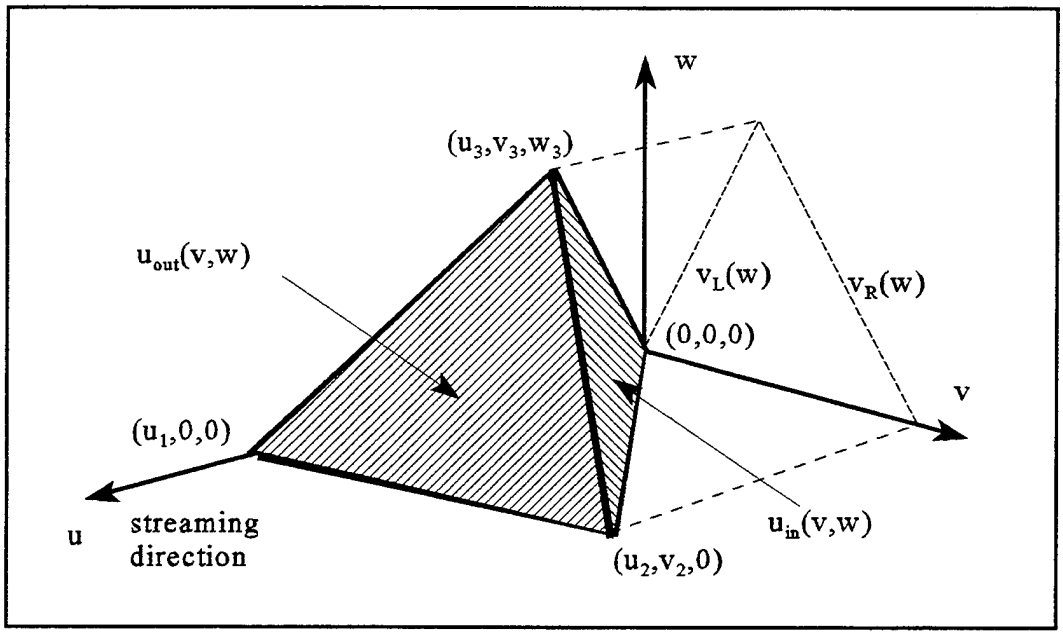

Figure 1. Case 1 Tetrahedron in the Local uvw Coordinate System,

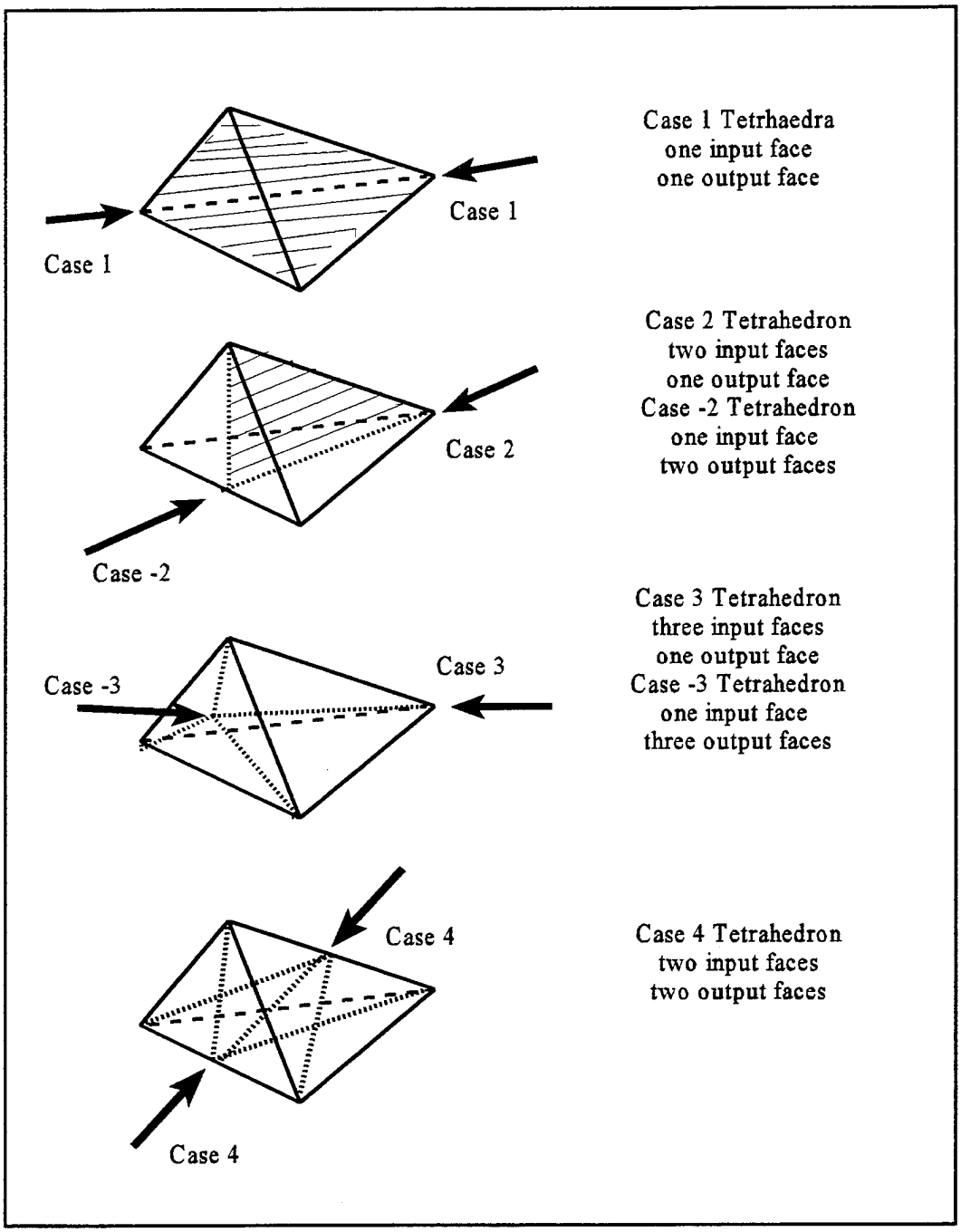

Figure 2. Tetrahedra Splitting and Resulting Number of Case 1 Sub-Tetrahedra.

3.3. Definition of "Arbitrary" Tetrahedra

When referring to meshes of arbitrary tetrahedra, the term "arbitrary" refers to the sizes, shape, and orientation of the tetrahedron in space and the characteristics of the individual tetrahedra. There is no restriction on how individual tetrahedra are oriented with respect to the defining coordinate system. There are no size or aspect ratio limits for the individual tetrahedra. Any number of tetrahedra may share an individual node. The only restriction is that each face must be shared by only two tetrahedra, i.e. faces may not be split so that a tetrahedron has more than a single neighbor across any given face.

3.4. Coordinate Systems

Like its triangle predecessor, the arbitrary tetrahedra algorithm computes such quantities as source distribution coefficients, entering flux distribution coefficients, and cell flux moments in local (orthogonal) coordinate systems designed to simplify both the derivation and resulting form of the moment equations. There are five separate coordinate systems used in the algorithm: global (x,y,z), centroid (x_c,y_c,z_c), face (x_F,y_F), source (u',v',w'), and local sub-tetrahedron (u,v,w).

The global coordinate system is generated with the problem geometry. All tetrahedra nodes are specified in the global xyz coordinate system. The global system remains fixed and independent throughout the problem execution. Each (parent) cell in the mesh has its own local, centroid coordinate system. For a given tetrahedron, the centroid system is generated by translating the origin of the global xyz

system to the tetrahedron centroid. All cell averaged quantities are computed in the centroid system. The relation between the global and centroid systems is shown in Figure 3.

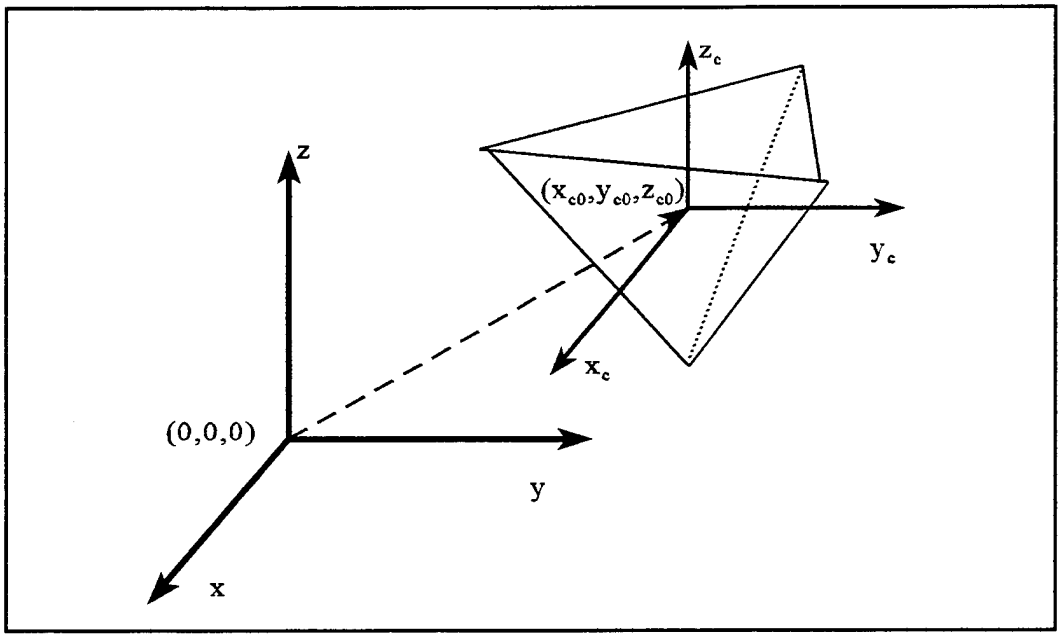

Figure 3. Orientation of Global and Centroid Coordinate Systems.

The face coordinate system, shown in Figure 4, is defined for each face in the mesh without regard to the tetrahedron with which the face is associated; i.e. the common face between two adjacent tetrahedra has a single coordinate system defined over it. This trivializes the passing of moments between adjacent tetrahedra: they are identical. For a given face, the system is defined so that the x_F axis lies along an edge and the apex not on the x_F axis has a positive y_F coordinate. The z_F axis is normal to the face, in the direction necessary to make a right-handed coordinate system. The selection of the apex defining

the origin and the edge defining the x_F axis are arbitrary, however once chosen, they remain fixed throughout the problem execution. Moments of the entering and exiting angular flux distribution are computed in terms of the face system for each tetrahedron. Note that if the x_F axis is chosen to be the longest edge of the triangle, $0 < \gamma < 1$, which may provide better numerical conditioning in the computation of the face distribution coefficients.

The source coordinate system is a local system defined for each original tetrahedron in the mesh, and is shown in Figure 5. It is used in the first order methods as a

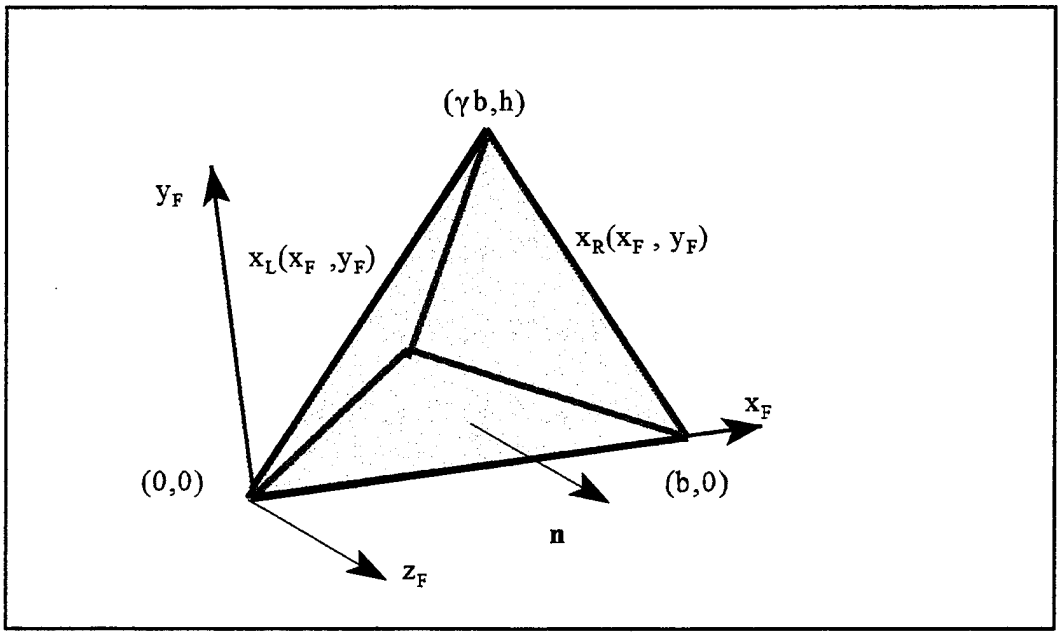

Figure 4. Global Face Coordinate System.

convenient system for computing the coefficients of the source distribution. The orientation of the tetrahedron in the source system is identical to the orientation of the

Case 1 tetrahedron in the local sub-tetrahedron coordinate system shown in Figure 1. The choice of the apex defining the system origin and the edge defining the u-axis are arbitrary, but once chosen, they remain fixed throughout the problem execution. Note that if the u-axis is chosen to run along the longest edge of the tetrahedron, then $u_1 > u_2 > 0$ and $u_1 > u_3 > 0$. This may provide better numerical conditioning in the computation of the source coefficients.

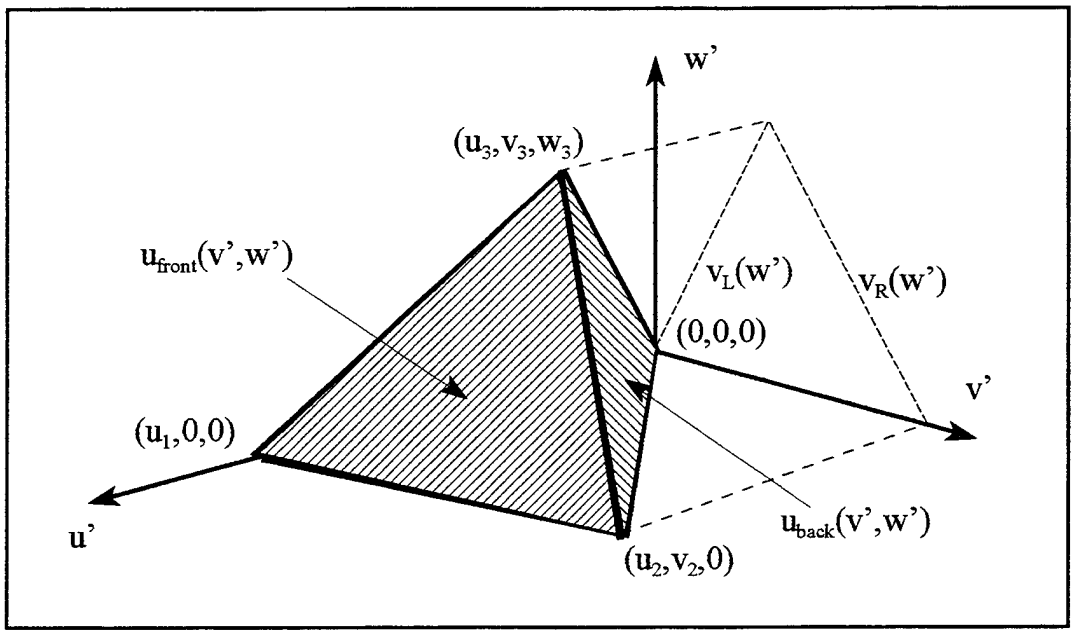

Figure 5. Coordinate System Used to Compute the Coefficients of the Source Distribution.

The local sub-tetrahedron uvw coordinate system was introduced in the previous section, and is shown in Figure 1. The spatial quadrature is applied in the local coordinate system, i.e. the source and entering flux distribution coefficients are transformed into the

local uvw system, and the cell and exiting flux moments are computed with respect to the local system for each case 1 tetrahedron.

3.5. Mathematical Preliminaries: The Exponential Moments Functions

The exponential moment functions are a class of functions originally developed to deal with removable singularities encountered in discrete ordinates methods. Numerical problems arise when the equations experience catastrophic cancellation as they approach their removable singularity. For example

$$\lim_{x \to 0} \left\{ \frac{\left(1 - e^{-x}\right)}{x} \right\} = 1 , \tag{35}$$

but when evaluated numerically as $x \to 0$, all precision is lost due to catastrophic cancellation in the numerator. When these types of singularities are encountered, the expression must be evaluated as a special case to preserve accuracy.

The first generation of moment functions (Walters,1986: 192-194; Walters, 1986: 22-24), were moment functions of a single argument. Mathews (Mathews, Sjonden, and Minor, 1995: 27) later developed a general integral definition of the functions and showed that it implied the recursion on order relationship. Minor (Minor, 1993: 46-59) extended the definition to include multiple arguments, and termed this broader class the exponential moment functions. In addition, Minor developed numerical routines to evaluate moment functions with up to three arguments. All but the single argument routines were rewritten

as part of this project to increase their computational efficiency. In addition, routines were developed to compute moment functions with four arguments.

Minor's dissertation (Minor, 1993) contains a complete description of the functions and the single argument algorithms. To facilitate both the derivation of the methods and the explanation of the moments function algorithm, this section begins with an overview of the functions.

3.5.1. Definition and Relationships

The exponential moment functions are defined as

$$M_n(x_1, x_2, \ldots, x_m) = \int_0^1 du_1 \int_0^{u_1} du_2 \ldots \int_0^{u_{m-1}} du_m \, (1-u)^n \, e^{-x_1 u_1} \, e^{(x_1 - x_2)u_2} \ldots e^{(x_{m-1} - x_m)u_m}, \qquad (36)$$

where n is termed the order of the function, and the dimensionality is determined by the number of arguments, m. Using this definition, an equation for the zeroth order function of a single argument is

$$M_0(x) = \frac{1 - e^{-x}}{x}. \qquad (37)$$

Given the above, higher order functions of a single argument can be found by recursion on order using

$$M_n(x) = \frac{1 - n M_{n-1}(x)}{x}. \qquad (38)$$

Recursion relationships also exist for functions of more than one variable, for example, any function of more than one variable can be constructed of lower dimension functions using recursion on dimension, given by

$$M_n(x_1,x_2,\ldots,x_m) = \frac{M_n(x_1,x_2,\ldots,x_{m-1}) - M_n(x_2,x_3,\ldots,x_m)}{x_m - x_1} \tag{39}$$

Functions of more than one argument with order greater than 0 can also be generated using recursion on order given by

$$M_n(x_1,x_2,\ldots,x_m) = \frac{M_n(x_1,x_2,\ldots,x_{m-1}) - nM_{n-1}(x_1,x_2,\ldots,x_m)}{x_m}. \tag{40}$$

Equation (39) is used in the numerical routines to compute moments functions when the values of the arguments are far enough apart to avoid numerical difficulties. Equation (40) is used in the moments functions routine to compute higher order functions having the same arguments.

The ratio of the first to zeroth order moment functions turns out to be a useful quantity for computing the flux moments in the EC method, hence, for simplicity of notation we define

$$\rho(x_1,x_2,\ldots,x_m) = \frac{M_1(x_1,x_2,\ldots,x_m)}{M_0(x_1,x_2,\ldots,x_m)}. \tag{41}$$

3.6. Derivation of the Characteristic Methods for Meshes of Arbitrary Tetrahedra

This section examines the derivation and application of the characteristic spatial quadratures for arbitrary tetrahedra. The spatial walk consists of the application of the spatial quadrature to each of the tetrahedra in the mesh. As the steps involved in applying the quadrature are exactly the same for each cell, we need only examine one. We begin with a discussion of the relatively simplistic SC method, explaining the steps required, derivation of the equations, and coordinate system transformations. The discussion then proceeds through the LC and EC methods, each of which introduces further complexity. After introducing the standard versions, we examine simplified variations of the LC and EC methods in which the source distributions retain their original form but the entering flux distribution is assumed constant.

3.6.1. Step Characteristic Method (SC)

The first step in developing a characteristic scheme is to specify the functional form of the source and entering flux distributions. For the SC method, both are assumed constant and equal to the average value of the distribution. The source distribution average is simply the zeroth spatial moment computed using equation (24). The entering flux average moment is given either by the boundary conditions or passed from the adjacent upstream cell(s). The averages are invariant with respect to coordinate transformations, hence they are identical for each sub-tetrahedron.

Given the source and entering flux moments, the tetrahedron is split along the streaming direction producing a number of sub-tetrahedra consistent with the case of the

parent. Local coordinate systems are then established for each subcell. The node coordinates are transformed from the global to the local uvw coordinate system using

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \mathbf{R} \begin{pmatrix} x - x_1 \\ y - y_1 \\ z - z_1 \end{pmatrix}, \tag{42}$$

where

$$\mathbf{R} = \begin{pmatrix} \hat{e}_u \cdot \hat{e}_x & \hat{e}_u \cdot \hat{e}_y & \hat{e}_u \cdot \hat{e}_z \\ \hat{e}_v \cdot \hat{e}_x & \hat{e}_v \cdot \hat{e}_y & \hat{e}_v \cdot \hat{e}_z \\ \hat{e}_w \cdot \hat{e}_x & \hat{e}_w \cdot \hat{e}_y & \hat{e}_w \cdot \hat{e}_z \end{pmatrix}, \tag{43}$$

and x_1, y_1, and z_1 are the coordinates of the uvw origin in the global coordinate system.

The moments of the subcell flux distribution are computed in the local coordinate system by substituting the constant forms of the source and entering flux distributions into equation (34) to get a functional form for $\psi(u,v,w)$ used in equation (26). The resulting integral can be separated into the source and entering flux contributions and the average flux moment is just their sum:

$$\psi_A^{subcell} = \psi_{A,S}^{subcell} + \psi_{A,in}^{subcell}. \tag{44}$$

The source contribution to the subcell flux moment is defined as

$$\psi_{A,S}^{subcell} = \left\langle \int_{u_{in}(v,w)}^{u} ds \; S_A \, e^{-\sigma(u-s)} \right\rangle_{subcell}, \tag{45}$$

where the subcell-volume integral operator, $\langle \cdot \rangle_{subcell}$, is defined as (reference Figure 1)

$$\langle \cdot \rangle_{cell} = \frac{1}{V} \int \cdot \, dV = 6 \int_0^{w_3} \frac{dw}{w_3} \int_{v_L(w)}^{v_R(w)} \frac{dv}{v_2} \int_{u_{in}(v,w)}^{u_{out}(v,w)} \frac{du}{u_1} \cdot, \tag{46}$$

where

$$\begin{aligned} u_{out}(v,w) &= c_1^{out} v + c_2^{out} w + u_1, \\ u_{in}(v,w) &= c_1^{in} v + c_2^{in} w, \\ v_L(w) &= \frac{v_3}{w_3} w \;, \; and, \\ v_R(w) &= \frac{(v_3 - v_2)}{w_3} w + v_2 . \end{aligned} \tag{47}$$

Additionally,

$$c_1^{out} = \frac{u_2 - u_1}{v_2},$$

$$c_2^{out} = \frac{v_2 u_3 - v_2 u_1 + u_1 v_3 - u_2 v_3}{v_2 w_3}, \tag{48}$$

$$c_1^{in} = \frac{u_2}{v_2}, \text{ and},$$

$$c_2^{in} = \frac{u_3 v_2 - u_2 v_3}{v_2 w_3}.$$

Evaluating equation (45) gives

$$\psi_{A,S}^{subcell} = u_1 S_A M_3(\epsilon) \tag{49}$$

where $\epsilon = \sigma u_1$ is the cell optical thickness.

The entering flux contribution to the subcell flux moment is defined as

$$\psi_{A,in}^{subcell} = \left\langle \psi_A^{in} e^{-\sigma\left[u - u_{in}(v,w)\right]} \right\rangle_{subcell}, \tag{50}$$

which, when evaluated gives

$$\psi_{A,in}^{subcell} = 3\, \psi_A^{in} M_2(\epsilon). \tag{51}$$

The moments of the exiting flux distribution are computed in the same way; substitute the constant forms of the source and entering flux distributions into equation (34) to get ψ(u,v,w) and use the result in equation (27). The total exiting flux moment can then be written as the sum of the source and exiting flux contributions:

$$\psi_A^{out} = \psi_{A,S}^{out} + \psi_{A,in}^{out} \,. \tag{52}$$

The source contribution to the exiting flux moment is defined as

$$\psi_{A,S}^{out} = \left\langle \int_{u_{in}(v,w)}^{u_{out}(v,w)} ds\, S_A\, e^{-\sigma[u_{out}(v,w)-s]} \right\rangle_{face}, \tag{53}$$

where the face integral operator, $\langle\cdot\rangle_{face}$, is defined as

$$\langle\cdot\rangle_{face} = \frac{1}{A}\int \cdot\, dA = 2\int_0^{w_3} \frac{dw}{w_3} \int_{v_L(w)}^{v_R(w)} \frac{dv}{v_2} \cdot \,. \tag{54}$$

Performing the integration yields the source contribution:

$$\psi_{A,S}^{out} = u_1\, S_A\, M_2(\epsilon) \,. \tag{55}$$

The entering flux contribution to the exiting flux moments is defined as

$$\psi_{A,in}^{out} = \left\langle \psi_A^{in} \, e^{-\sigma[u_{out}(v,w)-u_{in}(v,w)]} \right\rangle_{face}, \tag{56}$$

which, when evaluated gives the entering flux contribution:

$$\psi_{A,in}^{out} = 2\,\psi_A^{in} M_1(\epsilon) \,. \tag{57}$$

When the subcell moments are computed, they must be assembled to generate the parent cell results. As already stated, the average moments are invariant with respect to coordinate transformations, hence they have the same value in the centroid and global face systems as in the uvw system, and no transformation is necessary. The parent cell average flux is simply the volume weighted average of the subcell average fluxes:

$$\psi_A^{subcell} = \frac{1}{V} \sum_{i=1}^{\#\,subcells} V_i (\psi_A^{subcell})_i \,, \tag{58}$$

where V is the parent cell volume, V_i is the subcell volume, and $(\psi_A^{subcell})_i$ is the cell average flux moment for the i^{th} subcell. The subcell exiting flux moment is assembled as an area weighted average of the subface values:

$$\psi_A^{out} = \frac{1}{A} \sum_{i=1}^{\#\,subfaces} A_i (\psi_A^{out})_i \tag{59}$$

where A is the parent exiting face area, A_i is the sub-face area, and $(\psi_A^{out})_i$ is the zeroth exiting flux moment for the i^{th} sub-face.

3.6.2. Linear Characteristic Method (LC)

The LC method introduces further complexity because it carries first moments of each distribution. The first step in applying the LC spatial quadrature is to determine the coefficients of the source distribution. A convenient coordinate system for computing the source coefficients, because it simplifies the form of the moments integrals, is shown in Figure 5 (note that although the orientation is similar to the local case 1 coordinate system, the source system is defined for the *parent* tetrahedron).

The source moments are known either from initial conditions or from having been computed from the previous iteration scalar flux values. The moments must be transformed from the centroid system to the source system using

$$\begin{pmatrix} S_{u'} \\ S_{v'} \\ S_{w'} \end{pmatrix} = \mathbf{R}\left[\begin{pmatrix} S_{x_c} \\ S_{y_c} \\ S_{z_c} \end{pmatrix} - \begin{pmatrix} x_1 - x_{c0} \\ y_1 - y_{c0} \\ z_1 - z_{c0} \end{pmatrix} S_A\right], \tag{60}$$

where **R** is the rotation matrix constructed in the same way as equation (43), x_{c0}, y_{c0}, and z_{c0} are the coordinates of the centroid in the global system, and x_1, y_1, and z_1 are the coordinates of the source system origin in the global system (since all nodes are defined in the global system, the node defining the source system origin must first be translated to the centroid system). In the source system, the source moments are defined as

$$\begin{pmatrix} S_A \\ S_{u'} \\ S_{v'} \\ S_{w'} \end{pmatrix} = \left\langle \begin{pmatrix} 1 \\ u' \\ v' \\ w' \end{pmatrix} S(u',v',w') \right\rangle_{source}, \tag{61}$$

where the source volume integral operator, $\langle \cdot \rangle_{source}$, has the same form as the subcell volume integral operator given in equation (46), and $S(u',v',w') = a' + b'u' + c'v' + d'w'$. Evaluating the integrals gives

$$S_A = a' + \frac{b'}{4}(u_1 + u_2 + u_3) + \frac{c'}{4}(v_2 + v_3) + \frac{d'}{4}w_3$$

$$S_{w'} = \frac{a'}{4}w_3 + \frac{b'}{20}(u_1 + u_2 + 2u_3)w_3 + \frac{c'}{20}(v_2 + 2v_3)w_3 + \frac{d'}{10}w_3^2$$

$$\begin{aligned} S_{v'} = {} & \frac{a'}{4}(v_2 + v_3) + \frac{b'}{20}(u_1v_2 + 2u_2v_2 + u_3v_2 + u_1v_3 + u_2v_3 + 2u_3v_3) \\ & + \frac{c'}{10}(v_2^2 + v_2v_3 + v_3^2) + \frac{d'}{20}(v_2 + 2v_3)w_3 \end{aligned} \tag{62}$$

$$\begin{aligned} S_{u'} = {} & \frac{a'}{4}(u_1 + u_2 + u_3) + \frac{b'}{10}(u_1^2 + u_1u_2 + u_2^2 + u_1u_3 + u_2u_3 + u_3^2) \\ & + \frac{c'}{20}(u_1v_2 + 2u_2v_2 + u_3v_2 + u_1v_3 + u_2v_3 + 2u_3v_3) \\ & + \frac{d'}{20}(u_1 + u_2 + 2u_3)w_3 \end{aligned}$$

Equations (62) define a system of four equations in the four unknowns a', b', c', and d' which are analytically inverted to give a set of equations that is used to directly compute the coefficients given the source moments.

Once computed in the source system, the *coefficients* are transformed to the centroid system using

$$\begin{pmatrix} b_c \\ c_c \\ d_c \end{pmatrix} = \mathbf{R}^{\mathrm{T}} \begin{pmatrix} b' \\ c' \\ d' \end{pmatrix}, \tag{63}$$

and,

$$a_c = a' - b' t_1 - c' t_2 - d' t_3 , \tag{64}$$

where **R** is the rotation matrix used to transform the moments to the source system and

$$\begin{pmatrix} t_1 \\ t_2 \\ t_3 \end{pmatrix} = \mathbf{R} \begin{pmatrix} x_1 - x_{c0} \\ y_1 - y_{c0} \\ z_1 - z_{c0} \end{pmatrix} . \tag{65}$$

The coefficients in the centroid system can then be transformed into the local sub-tetrahedra systems as needed. This is more computationally efficient than transforming the source moments into the subcell systems and directly computing the coefficients of the distribution for each subcell.

We use a similar approach to determine the coefficients of the entering flux distribution. The moments are known in the global face system; either from boundary conditions or from upstream cells. In the global face system, the moments of the entering flux distribution are defined as

$$\begin{pmatrix} \psi_A^{in} \\ \psi_x^{in} \\ \psi_y^{in} \end{pmatrix} = \left\langle \begin{pmatrix} 1 \\ x_F \\ y_F \end{pmatrix} \psi^{in}(x_F, y_F) \right\rangle_{face}, \tag{66}$$

where $\psi^{in}(x_F, y_F) = \alpha + \beta_x x_F + \beta_y y_F$, and the entering face integral operator, $\langle \cdot \rangle_{face}$, is defined as

$$\langle \cdot \rangle_{face} = \frac{1}{A_{face}} \int \cdot \, dA_{face} = 2 \int_0^h \frac{dy_F}{h} \int_{x_L(y_F)}^{x_R(y_F)} \frac{dx_F}{b} \cdot, \tag{67}$$

where $x_L(y_F) = \frac{\gamma b}{h} y_F$, and, $x_R(y_F) = b - \frac{b(1-\gamma)}{h} y_F$. Evaluating the integrals gives

$$\begin{aligned} \psi_A^{in} &= \frac{1}{3}\left[3\alpha + b\beta_x(1+\gamma) + h\beta_y\right] \\ \psi_y^{in} &= \frac{h}{12}\left[4\alpha + b\beta_x(1+2\gamma) + 2h\beta_y\right] \\ \psi_x^{in} &= \frac{b}{12}\left[4\alpha(1+\gamma) + 2b\beta_x(1+\gamma+\gamma^2) + 2h\beta_y(1+2\gamma)\right]. \end{aligned} \tag{68}$$

Equations (68) comprise a linear system of three equations in the three unknowns α, β_x, and β_y. Solving this system analytically gives a system of equations for directly computing the coefficients given the entering flux moments.

Once the coefficients of the source and entering flux coefficients are known, they can be transformed into the local uvw system to use in computing the subcell moments. For the source coefficients:

$$\begin{pmatrix} b \\ c \\ d \end{pmatrix} = \mathbf{R} \begin{pmatrix} b_c \\ c_c \\ d_c \end{pmatrix}, \tag{69}$$

and $a = a_c + b t_1 + c t_2 + d t_3 - b_c x_{c0} - c_c y_{c0} - d_c z_{c0}$. The entering flux coefficients are transformed from the global face system to the local uvw system using

$$\begin{pmatrix} \beta_v \\ \beta_w \end{pmatrix} = \mathbf{R}_{\mathbf{face}} \begin{pmatrix} \beta_x \\ \beta_y \end{pmatrix}, \tag{70}$$

and $\alpha_{vw} = \alpha - \beta_v t_1 - \beta_w t_2$, where in this case, $(t_1, t_2)^{\mathbf{T}}$ is the translation vector from the global face to the local uvw origin (the translation can be done in two dimensions because the origins of both systems are in the same face).

As in the SC method, the subcell flux moments for the LC method are the sum of the source and entering flux contributions:

$$\begin{aligned}
\psi_A^{subcell} &= \psi_{A,in}^{subcell} + \psi_{A,S}^{subcell} \\
\psi_u^{subcell} &= \psi_{u,in}^{subcell} + \psi_{u,S}^{subcell} \\
\psi_v^{subcell} &= \psi_{v,in}^{subcell} + \psi_{v,S}^{subcell} \\
\psi_w^{subcell} &= \psi_{w,in}^{subcell} + \psi_{w,S}^{subcell} \quad .
\end{aligned} \tag{71}$$

The source contributions to the cell flux moments are defined as

$$\begin{pmatrix} \psi_{A,S}^{subcell} \\ \psi_{u,S}^{subcell} \\ \psi_{v,S}^{subcell} \\ \psi_{w,S}^{subcell} \end{pmatrix} = \left\langle \int_{u_{in}(v,w)}^{u} ds \begin{pmatrix} 1 \\ u \\ v \\ w \end{pmatrix} S(s,v,w)\ e^{-\sigma(u-s)} \right\rangle_{subcell}, \tag{72}$$

where $S(s,v,w) = a + bs + cv + dw$ and the subcell-volume integral operator was defined in equation (46). Evaluating the integrals gives

$$\psi_{A,S}^{subcell} = a u_1 M_3(\epsilon) + \frac{u_1}{4}\left[b(u_1+u_2+u_3) + c(v_2+v_3) + d w_3 \right] M_4(\epsilon)$$

$$\psi_{w,S}^{subcell} = \frac{a u_1 w_3}{4} M_4(\epsilon) + \frac{u_1 w_3}{20}\left[b(u_1+u_2+2u_3) + c(v_2+2v_3) + 2d w_3 \right] M_5(\epsilon)$$

$$\begin{aligned}\psi_{v,S}^{subcell} = {} & \frac{a u_1 (v_2+v_3)}{4} M_4(\epsilon) + \frac{u_1}{4}\Big[b(v_2(u_1+2u_2+u_3) + v_3(u_1+u_2+2u_3)) \\ & + 2c(v_2^2 + v_2 v_3 + v_3^2) + d w_3 (v_2+2v_3) \Big] M_5(\epsilon)\end{aligned} \tag{73}$$

$$\begin{aligned}\psi_{u,S}^{subcell} = {} & a u_1^2 M_3(\epsilon) \\ & + \frac{u_1}{4}\left[a(-3u_1+u_2+u_3) + b u_1 (u_1+u_2+u_3) + c u_1 (v_2+v_3) + d u_1 w_3 \right] M_4(\epsilon) \\ & + \frac{u_1}{20}\Big[b(-3u_1^2 - 3u_1 u_2 + 2u_2^2 - 3u_1 u_3 + 2u_2 u_3 + 2u_3^2) \\ & \quad + c(-4u_1 v_2 + 2u_2 v_2 + u_3 v_2 - 4u_1 v_3 + u_2 v_3 + 2u_3 v_3) \\ & \quad + d w_3 (-4u_1 + u_2 + 2u_3) \Big] M_5(\epsilon) .\end{aligned}$$

The entering flux contributions to the cell flux moments are defined as

$$\begin{pmatrix} \psi_{A,in}^{subcell} \\ \psi_{u,in}^{subcell} \\ \psi_{v,in}^{subcell} \\ \psi_{w,in}^{subcell} \end{pmatrix} = \left\langle \begin{pmatrix} 1 \\ u \\ v \\ w \end{pmatrix} \psi^{in}[u_{in}(v,w), v, w]\; e^{-\sigma\left[u - u_{in}(v,w)\right]} \right\rangle_{subcell}, \tag{74}$$

where $\psi^{in}[u_{in}(v,w), v, w] = \alpha_{vw} + \beta_v v + \beta_w w$, yielding

$$\psi_{A,in}^{subcell} = 3\alpha_{vw} M_2(\epsilon) + \left[\beta_v(v_2+v_3)+\beta_w w_3\right] M_3(\epsilon)$$

$$\psi_{w,in}^{subcell} = \alpha_{vw} w_3 M_3(\epsilon) + \frac{w_3}{4}\left[\beta_v(v_2+2v_3)+2\beta_w w_3\right] M_4(\epsilon)$$

$$\psi_{v,in}^{subcell} = \alpha_{vw}(v_2+v_3) M_3(\epsilon) + \frac{1}{4}\left[\beta_v(2v_2^2+2v_2v_3+2v_3^2)+\beta_w w_3(v_2+2_{v_3})\right] \qquad (75)$$

$$\begin{aligned}\psi_{u,in}^{subcell} = {} & 3\alpha_{vw} u_1 M_2(\epsilon) + \left[\alpha(-3u_1+u_2+u_3)+\beta_v u_1(v_2+v_3)+\beta_w w_3 u_1\right] M_3(\epsilon) \\ & + \frac{1}{4}\left[\beta_v(2u_2v_2-4u_1v_2+u_3v_2-4u_1v_3+u_2v_3+2u_3v_3)\right. \\ & \left. + \beta_w w_3(-4u_1+u_2+2u_3)\right] M_4(\epsilon) .\end{aligned}$$

The moments of the exiting flux distribution are also computed in terms of the source and entering flux contributions:

$$\begin{aligned}\psi_A^{out} &= \psi_{A,S}^{out} + \psi_{A,in}^{out} \\ \psi_v^{out} &= \psi_{v,S}^{out} + \psi_{v,in}^{out} \\ \psi_w^{out} &= \psi_{w,S}^{out} + \psi_{w,in}^{out} .\end{aligned} \qquad (76)$$

The source contribution is defined as

$$\begin{pmatrix} \psi_{A,S}^{out} \\ \psi_{v,S}^{out} \\ \psi_{w,S}^{out} \end{pmatrix} = \left\langle \int_{u_{in}(v,w)}^{u_{out}(v,w)} ds \begin{pmatrix} 1 \\ v \\ w \end{pmatrix} S(s,v,w)\, e^{-\sigma[u_{out}(v,w)-s]} \right\rangle_{face}, \qquad (77)$$

where the face integral operator, $\langle \cdot \rangle_{face}$, was defined in equation (54). Evaluating the integrals gives the equations for the source contributions:

$$\begin{aligned}
\psi_{A,S}^{out} &= a u_1 M_2(\epsilon) + \frac{u_1}{3}\left[b(u_1+u_2+u_3) + c(v_2+v_3) + d w_3 \right] M_3(\epsilon) \\
\psi_{w,S}^{out} &= \frac{a u_1 w_3}{3} M_3(\epsilon) + \frac{u_1 w_3}{3}\left[b(u_1+u_2+2u_3) + c(v_2+2v_3) + 2d w_3 \right] M_4(\epsilon) \\
\psi_{v,S}^{out} &= \frac{a u_1 (v_2+v_3)}{3} M_3(\epsilon) \\
&\quad + \frac{u_1}{12}\left[b\left[(v_2+v_3)(u_1+u_2+u_3) + u_2 v_2 + u_3 v_3 \right] \right. \\
&\quad \left. + c(2v_2^2 + 2v_2 v_3 + 2v_3^2) + d w_3 (v_2+2v_3) \right] M_4(\epsilon) .
\end{aligned} \tag{78}$$

The entering flux contributions to the exiting flux moments are defined as

$$\begin{pmatrix} \psi_{A,in}^{out} \\ \psi_{v,in}^{out} \\ \psi_{w,in}^{out} \end{pmatrix} = \left\langle \begin{pmatrix} 1 \\ v \\ w \end{pmatrix} \psi^{in}[u_{in}(v,w), v, w]\; e^{-\sigma[u_{out}(v,w) - u_{in}(v,w)]} \right\rangle_{face}, \tag{79}$$

which, when evaluated give:

$$\begin{aligned}
\psi_{A,in}^{out} &= 2\alpha M_1(\epsilon) + \left[\beta_v (v_2+v_3) + \beta_w w_3 \right] M_2(\epsilon) \\
\psi_{w,in}^{out} &= \alpha w_3 M_2(\epsilon) + \frac{w_3}{3}\left[\beta_v (v_2+2v_3) + 2\beta_w w_3 \right] M_3(\epsilon) \\
\psi_{v,in}^{out} &= \alpha (v_2+v_3) M_2(\epsilon) + \frac{1}{3}\left[\beta_v (2v_2^2 + 2v_2 v_3 + 2v_3^2) + \beta_w w_3 (v_2+2v_3) \right] M_3(\epsilon) .
\end{aligned} \tag{80}$$

The subcell moments must be transformed back to the parent cell coordinate systems before they can be assembled into the parent cell results. The cell flux first moments are rotated back to the centroid coordinate system using

$$\begin{pmatrix} \psi^i_{xc} \\ \psi^i_{yc} \\ \psi^i_{zc} \end{pmatrix} = \mathbf{R}_i^{\mathrm{T}} \begin{pmatrix} \psi^i_{u} \\ \psi^i_{v} \\ \psi^i_{w} \end{pmatrix} + \begin{pmatrix} x_1 - x_{c0} \\ y_1 - y_{c0} \\ z_1 - z_{c0} \end{pmatrix} \psi^i_A \,, \tag{81}$$

where $\mathbf{R}_i$ is the rotation matrix for the i^{th} sub-tetrahedron. The exiting face moments are transformed using

$$\begin{pmatrix} \psi^{out}_{xF} \\ \psi^{out}_{yF} \end{pmatrix} = \mathbf{R} \begin{pmatrix} \psi^{out}_{v} \\ \psi^{out}_{w} \end{pmatrix} + \begin{pmatrix} t_1 \\ t_2 \end{pmatrix} \psi^{out}_A \tag{82}$$

where $(t_1, t_2)^{\mathrm{T}}$ is the translation vector from the local uvw origin to the face system origin.

In the centroid system, the subcell flux moments have to be assembled into the parent cell results. As they were for SC, the parent cell moments are just the volume weighted average of the subcell moments:

$$\begin{pmatrix} \psi_A \\ \psi_{xc} \\ \psi_{yc} \\ \psi_{zc} \end{pmatrix} = \frac{1}{V} \sum_{i=1}^{|\#\ subcells|} V_i \begin{pmatrix} \psi^i_A \\ \psi^i_{xc} \\ \psi^i_{yc} \\ \psi^i_{zc} \end{pmatrix} , \tag{83}$$

where V is the parent cell volume, V_i is the subcell volume, and ψ^i is the flux moment for the i^{th} subcell. Likewise, the exiting flux moments for the parent cell are just an area weighted average of the subcell results:

$$\begin{pmatrix} \psi_A^{out} \\ \psi_{xF}^{out} \\ \psi_{yF}^{out} \end{pmatrix} = \frac{1}{A} \sum_{i=1}^{|\#subfaces|} A_i \begin{pmatrix} \psi_A^{out,i} \\ \psi_{x_F}^{out,i} \\ \psi_{y_F}^{out,i} \end{pmatrix} \tag{84}$$

where A is the parent exiting face area and A_i is the subcell face area.

3.6.3. Exponential Characteristic Method (EC)

The implementation of the EC method is similar to that of the LC method in that both carry first moments of the distribution so the application of the quadrature includes the same sequence of steps. There are, however, several significant differences. In the equations defining the moment equations for the LC method, the only exponential term came from the attenuation of the entering flux and source distributions, and the integrations produced exponential moment functions of a single dimension. The EC distributions are defined in terms of exponentials, so the equations defining the moments generate multi-dimensional exponential moment functions. Further, the systems of equations for the source and entering flux moments are nonlinear, and the distribution coefficients must be found using Newton's method.

As in the LC method, the first step in applying the EC quadrature is to transform the source moments to the source coordinate system. This is accomplished using equation (60). In the source coordinate system, the moments of the source distribution are defined in equation (61) where $S(u',v',w') = a' e^{b'u'+c'v'+d'w'}$, which when evaluated gives

$$\begin{aligned} S_A &= 6a' e^{Y} M_0(X,Y,Z) \\ S'_w &= S_A \, w_3 \, \rho(X,Y,Z) \\ S'_v &= S_A \left[v_3 \, \rho(X,Y,Z) + v_2 \, \rho(Y-X,Z-X,-X) \right] \\ S'_u &= S_A \left[u_3 \, \rho(X,Y,Z) + u_2 \, \rho(Y-X,Z-X,-X) + u_1 \, \rho(Y-Z,X-Z,-Z) \right], \end{aligned} \tag{85}$$

where

$$\begin{aligned} Y &= b'u_3 + c'v_3 + d'w_3, \\ X &= Y - b'u_2 - c'v_2, \\ Z &= Y - b'u_1. \end{aligned} \tag{86}$$

As in the LC method, the system defined by equations (85) can be inverted to find the distribution coefficients. However, unlike LC, this system is non-linear and must be solved using some root solving method such as Newton's Method. Appendix B contains a complete discussion of the root-solving procedure. Once computed in the source system, the coefficients are rotated back to the centroid system using equation (63) and $a_c = a' e^{-b't_1 - c't_2 - d't_3}$ where they are "stored" until needed for the subcell computations.

The moments of the entering flux distribution for the EC method are defined in equation (66) where $\psi^{in}(x_F, y_F) = \alpha e^{\beta_x x_F + \beta_y y_f}$. Evaluating the integrals gives

$$\begin{aligned}
\psi_A^{in} &= 2\alpha e^{X_{in}} M_0(X_{in}, Y_{in}) \\
\psi_y^{in} &= \psi_A^{in} h\, \rho(X_{in}, Y_{in}) \\
\psi_x^{in} &= \psi_A^{in} \left[\gamma h\, \rho(X_{in}, Y_{in}) + b\, \rho(X_{in} - Y_{in}, -Y_{in})\right].
\end{aligned} \tag{87}$$

where $X_{in} = h\beta_y + \gamma b \beta_x$ and $Y_{in} = X_{in} - b\beta_x$.

Equations (87) define a non-linear system in the constants X_{in} and Y_{in} that are solved using Newton's method. Appendix B contains a complete discussion of the two-dimensional root solving procedure.

Once computed, the source and entering flux coefficients are transformed to the local uvw system. For the source moments, the first moments are rotated exactly as the LC moments, using equation (63). The translation is accounted for in the constant term, which in this case is $a = a_c e^{bt_1 + ct_2 + dt_3 - b_c x_{c0} - c_c y_{c0} - d_c z_{c0}}$. The entering flux first moments are rotated using equation (70) and the translation is accounted for in $\alpha_{vw} = \alpha e^{\beta_v t_1 - \beta_w t_2}$.

The source and entering flux contributions to the subcell flux moments are given by equations (72) and (74) where the exponential forms of the distributions are used:

$$\begin{aligned}
S(s, v, w) &= a e^{bs + cv + dw} \\
\psi^{in}[u_{in}(v, w), v, w] &= \alpha_{vw} e^{\beta_v v + \beta_w w}.
\end{aligned} \tag{88}$$

Evaluating the integrals gives the equations for computing the source contributions,

$$
\begin{aligned}
\psi_{A,S}^{subcell} &= 6\,a\,u_1\,e^{Y_s} M_0(X_s, Y_s, Z_s, \Gamma_s) \\
\psi_{A,S}^{subcell} &= \psi_{A,S}^{subcell}\left[\, w_3\,\rho(X_s, Y_s, Z_s, \Gamma_s)\,\right] \\
\psi_{v,S}^{subcell} &= \psi_{A,S}^{subcell}\left[\, v_3\,\rho(X_s, Y_s, Z_s, \Gamma_s) + v_2\,\rho(\Gamma_s - X_s, Y_s - X_s, Z_s - X_s, -X_s)\,\right] \\
\psi_{u,S}^{subcell} &= \psi_{A,S}^{subcell}\,[\, u_3\,\rho(X_s, Y_s, Z_s, \Gamma_s) + u_2\,\rho(\Gamma_s - X_s, Y_s - X_s, Z_s - X_s, -X_s) \\
&\qquad + u_1\,\rho(\Gamma_s - Z_s, Y_s - Z_s, X_s - Z_s, -Z_s) + u_1\,\rho(Z_s - \Gamma_s, Y_s - \Gamma_s, X_s - \Gamma_s, -\Gamma_s)\,]
\end{aligned}
\tag{89}
$$

where

$$
\begin{aligned}
Y_s &= b u_3 + c v_3 + d w_3 \\
X_s &= Y_s - b u_2 - c v_2 \\
Z_s &= Y_s - b u_1 \\
\Gamma_s &= Y_s + \epsilon\,,
\end{aligned}
\tag{90}
$$

and the equations for computing the entering flux contributions,

$$
\begin{aligned}
\psi_{A,in}^{subcell} &= 6\,a\,e^{Y_{inc}} M_0(X_{inc}, Y_{inc}, Z_{inc}) \\
\psi_{w,in}^{subcell} &= \psi_{A,in}^{subcell}\left[\, w_3\,\rho(X_{inc}, Y_{inc}, Z_{inc})\,\right] \\
\psi_{v,,in}^{subcell} &= \psi_{A,in}^{subcell}\left[\, v_3\,\rho(X_{inc}, Y_{inc}, Z_{inc}) + v_2\,\rho(Y_{inc} - X_{inc}, Z_{inc} - X_{inc}, -X_{inc})\,\right] \\
\psi_{u,in}^{subcell} &= \psi_{A,in}^{subcell}\,[\, u_3\,\rho(X_{inc}, Y_{inc}, Z_{inc}) + u_2\,\rho(Y_{inc} - X_{inc}, Z_{inc} - X_{inc}, -X_{inc}) \\
&\qquad + u_1\,\rho(Y_{inc} - Z_{inc}, X_{inc} - Z_{inc}, -Z_{inc})\,]
\end{aligned}
\tag{91}
$$

where

$$
\begin{aligned}
X_{inc} &= Y_{in} \\
Y_{inc} &= X_{inc} + \beta_v v_2 \\
Z_{inc} &= Y_{inc} + \epsilon \, .
\end{aligned}
\tag{92}
$$

The equations for computing the source and entering flux contributions to the exiting flux moments for the EC method are generated by substituting the exponential forms of the distributions, equations (32), into equations (77) and (79). This gives the source contribution equations,

$$
\begin{aligned}
\psi_{A,S}^{out} &= 2u_1 a e^{Y_s} M_0(X_s, Z_s, \Gamma_s) \\
\psi_{w,S}^{out} &= \psi_{A,S}^{out}\left[w_3 \rho(X_s, Z_s, \Gamma_s) \right] \\
\psi_{v,S}^{out} &= \psi_{A,S}^{out}\left[v_3 \rho(X_s, Z_s, \Gamma_s) + v_2 \rho(Z_s - X_s, \Gamma_s - X_s, -X_s) \right] .
\end{aligned}
\tag{93}
$$

and the entering flux contribution equations,

$$
\begin{aligned}
\psi_{A,in}^{out} &= 2\alpha e^{Y_{inc}} M_0(X_{inc}, Z_{inc}) \\
\psi_{w,in}^{out} &= \psi_{A,in}^{out}\left[w_3 \rho(X_{inc}, Z_{inc}) \right] \\
\psi_{v,in}^{out} &= \psi_{A,in}^{out}\left[v_3 \rho(X_{inc}, Z_{inc}) + v_2 \rho(Z_{inc} - X_{inc}, -X_{inc}) \right] .
\end{aligned}
\tag{94}
$$

The remaining steps, transforming the subcell flux moments to the centroid system, transforming the subcell exiting flux moments to the face system, assembling the parent cell results, and the equations used to accomplish them, are identically the same for both the LC and EC method.

3.6.4. Constant Face Variants

Two additional methods were derived and tested: Constant Linear Characteristic (CLC) and Constant Exponential Characteristic (CEC). These methods retain the assumed form of the source distribution used in the LC and EC methods, but assume a constant distribution on the face.

3.6.4.1. Constant Linear Characteristic

The general flow of the CLC method is the same as that for the methods already presented. First, the coefficients of the source moments have to be determined. Both the LC and CLC methods use the same assumed form of the source distribution, so the procedure for computing the CLC source distribution coefficients is identical to the LC method. The entering flux distribution is assumed constant, hence it is handled the same way as the SC method.

In the local coordinate system, the source contributions to the subcell flux moments are computed as in the LC method, using equations (72). The equations for computing the entering flux contributions to the subcell flux are generated by substituting the constant form of the entering flux distribution into equation (74) giving

$$\psi_{A,in}^{subcell} = 3\psi_A^{in} M_2(\epsilon)$$

$$\psi_{w,in}^{subcell} = \psi_A^{in} w_3 M_3(\epsilon)$$

$$\psi_{v,in}^{subcell} = \psi_A^{in}(v_2 + v_3) M_3(\epsilon) \qquad (95)$$

$$\psi_{u,in}^{subcell} = 3\psi_A^{in} u_1 M_2(\epsilon) + \psi_A^{in} M_3(\epsilon)(-3u_1 + u_2 + u_3) .$$

The exiting face distribution is constant, only the zeroth moment is needed. The source contribution to the exiting flux moment is computed using the LC source contribution to the exiting face average moment in equation (78),

$$\psi_{A,S}^{out} = a u_1 M_2(\epsilon) + \frac{u_1}{3}\left[b(u_1 + u_2 + u_3) + c(v_2 + v_3) + d w_3\right] M_3(\epsilon) . \qquad (96)$$

The entering flux contribution to the exiting flux moment is the same as for the SC method:

$$\psi_{A,in}^{out} = 2\psi_A^{in} M_1(\epsilon) . \qquad (97)$$

3.6.4.2. Constant Exponential Characteristic

The source moments and coefficient calculations are the same for the EC and CEC methods. The entering flux moment for the EC method is handled the same way as the SC method. In the local coordinate system, the source contributions to the cell flux moments

are again computed using equations (89). Because the entering flux is assumed constant, the entering flux contributions to the cell flux moments are the same as for CLC; equations (95). The source contribution to the exiting flux moment is the same as the EC source contribution to the exiting flux average moment from equation (93),

$$\psi_{A,S}^{out} = 2u_1 a e^{Y_s} M_0(X_s, Z_s, \Gamma_s) , \tag{98}$$

and the entering flux contribution to the exiting flux moment is identical to that used in the SC and CLC methods.

3.7. Balance Equations

The balance equations are obtained by taking moments of the Boltzmann transport equation over the case 1 cell,

$$\left\langle \begin{pmatrix} 1 \\ u \\ v \\ w \end{pmatrix} \left[\frac{\partial}{\partial u} \psi(u,v,w) + \sigma(u,v,w)\, \psi(u,v,w) = S(u,v,w) \right] \right\rangle_{cell} , \tag{99}$$

where the volume integration is carried out over the case 1 volume. Using equation (98), the zeroth-moment equation is

$$3(\psi_A^{out} - \psi_A^{in}) + \epsilon \psi^A = u_1 S^A , \tag{100}$$

the w-moment equation is

$$3(\psi_w^{out} - \psi_w^{in}) + \epsilon\psi^w = u_1 S^w , \tag{101}$$

and the v-moment equation is,

$$3(\psi_v^{out} - \psi_v^{in}) + \epsilon\psi^v = u_1 S^v . \tag{102}$$

To derive the u-Moment equation refer to Figure 1 and recall that on the entering face,

$$u_{in}(v,w) = c_1^{in} v + c_2^{in} w , \tag{103}$$

and on the exiting face,

$$u_{out}(v,w) = c_1^{out} v + c_2^{out} w + u_1 , \tag{104}$$

where

$$c_1^{in} = \frac{u_2}{v_2} , \; c_2^{in} = \frac{u_3 v_2 - u_2 v_3}{v_2 w_3} ,$$
$$c_1^{out} = \frac{u_2 - u_1}{v_2}, \quad \text{and} \quad c_2^{out} = \frac{v_2(u_3 - u_1) + v_3(u_1 - u_2)}{v_2 w_3} . \tag{105}$$

The u-Moment equation is then

$$3(c_1^{out}\psi_v^{out} + c_2^{out}\psi_w^{out}) - 3(c_1^{in}\psi_v^{in} + c_2^{in}\psi_w^{in}) + 3u_1\psi_A^{out} - u_1\psi^A + \epsilon\psi^u = u_1 S^u . \tag{106}$$

Using equations (103) and (104),

$$\psi_u^{in} = c_1^{in}\psi_v^{in} + c_2^{in}\psi_w^{in}$$
$$\psi_u^{out} = c_1^{out}\psi_v^{out} + c_2^{out}\psi_w^{out} + u_1\psi_A^{out} \qquad (107)$$

and the u-Moment equation can now be written in a form consistent with the other conservation equations as

$$3(\psi_u^{out} - \psi_u^{in}) - u_1\psi_A + \epsilon\psi^u = u_1 S^u . \qquad (108)$$

The balance, or conservation equations serve several purposes. They serve as a check of the analytic results of the derivations. The analytic forms of the moment equations for each of the quadratures were checked by substituting them into the balance equations. During execution, after the cell and exiting flux moments are computed for a sub-tetrahedron, the results are checked by substituting the numerical results into the balance equations. This ensures that no loss of digits due to errors or numerical ill-conditioning corrupts the solution. Alternatively, the equations can be used as a more efficient means of computing the cell flux moments.

Computing the cell flux moments for the EC method using equations (89) and (91) is very expensive. Using the conservation equations in the form

$$\psi_A = \frac{u_1 S_A + 3\psi_A^{in}}{\epsilon} - \frac{3\psi_A^{out}}{\epsilon}$$

$$\psi_w = \frac{u_1 S_w + 3\psi_w^{in}}{\epsilon} - \frac{3\psi_w^{out}}{\epsilon} \qquad (109)$$

$$\psi_v = \frac{u_1 S_v + 3\psi_v^{in}}{\epsilon} - \frac{3\psi_v^{out}}{\epsilon}$$

$$\psi_u = \frac{u_1 S_u + u_1 \psi_A + 3\psi_u^{in}}{\epsilon} - \frac{3\psi_u^{out}}{\epsilon}$$

is a much more efficient way to generate the cell flux moments, since all of the quantities on the right hand side of the equations are either input to the quadrature (S_A, S_u, S_v, S_w), computed as part of the quadrature $(\psi_A^{out}, \psi_v^{out}, \psi_w^{out})$, or would have been computed anyway to check conservation on the subcell $(\psi_A^{in}, \psi_v^{in}, \psi_w^{in})$ (since only the coefficients of the entering flux distribution are transformed to the subcell, the moments over the subcell inflow face must be constructed from the fitted distribution). Equations (109) become numerically ill-conditioned as the cell optical thickness approaches zero, and they must be used with care. Further details on the use of the conservation equations to compute the cell moments can be found in Appendix D.

IV. Arbitrary Tetrahedra Code Algorithm

The characteristic methods were implemented in the TetSN code developed as part of this research. The TetSN algorithm is a direct extension of Miller's algorithm (Miller, 1993) to three-dimensions.

4.1. Initialization

When executed, the code first reads runtime data from a file containing the names of the input and output files, the order of the angular quadrature, the spatial quadrature to use, moments function computation tolerance, and the problem convergence tolerance. The input file contains all information pertaining to mesh and materials. It consists of a list of apex triplets (the xyz coordinate of each apex in the global coordinate system), a list of node numbers associated with each tetrahedron, the material number of each tetrahedron, the material cross section and source information for each material and energy group, the group-to-group scattering cross sections, and the boundary conditions. The code then generates lists of neighbor tetrahedra, boundary tetrahedra, boundary faces, face areas, cell volumes, and centroid locations. Given the angular quadrature order, the code fills the direction cosine and weight arrays and sets the number of angles. The highest energy group scalar fluxes are then initialized either to an a priori estimate or a default value of zero.

4.2. Iteration

Assuming no up scatter (particles may not gain energy in collisions), the group source moments are computed using equation (28). Each energy group uses all of the

higher-energy-group-scalar fluxes to compute its scattering source. Therefore, the program execution consists of the outer iteration, the iteration on the between the group scattering source, and the inner iteration, the iteration on the within group scattering source. The outer iteration proceeds through each of the energy groups, beginning with the highest energy group. The scalar flux for that group is converged (the inner iteration) and then used to update the down-scatter source for each of the lower-energy groups. The iteration then proceeds down through the rest of the groups. Note that for down scatter only, the outer iteration simply consists of a single pass through each of the energy groups. If up scatter were allowed, the lower group fluxes would contribute to the scattering source for higher energy groups, and the outer iteration would have to be repeated until the group fluxes all converged.

For the first order methods, the coefficients of the source distribution must be computed for each tetrahedron. Because the scatter is assumed to be isotropic, the source moments are independent of angle, and the source moment computation need only be performed once per iteration per tetrahedron. The scalar flux moments and hence the source moments are computed with respect to the centroid in the centroid coordinate system, and transformed to the local uvw coordinate system where the coefficients are determined. Once computed in the source coordinate system, either by a linear system of equations for LC and CLC or by Newton's method for EC and CEC, the source distribution coefficients are transformed back to the centroid coordinate system where they are "stored" until needed.

At the start of each inner iteration, the code saves the previous iteration scalar fluxes to use to check for convergence at the end of the current iteration. Before each spatial walk, the code resets all cell angular fluxes to zero and sets face angular fluxes in accordance with the boundary conditions. It then computes the dot product between the streaming direction and the outward normal of every face of every tetrahedron to determine whether the face is input or output. A push-down stack is used to store the indices of cells for which all necessary input data are available. Every tetrahedron in the mesh is compared to the input data list to determine which tetrahedra have the necessary input information. Tetrahedra having input information for each input face are appended to the stack, and the stack pointer is incremented.

After all the tetrahedra have been checked for input data, the last tetrahedron is pulled from the stack, and the stack pointer decremented. The entering flux distribution coefficients for each input face are then computed using the equations derived in Chapter 3. As with the source moments, the coefficients of the entering flux distribution must be generated from knowledge of the zeroth and first moments. All face flux moments are computed in terms of a face coordinate system. This trivializes the passing of moments between adjacent tetrahedra. Since they are defined in the same coordinate system, the entering face moments of one tetrahedron are identically equal to the exiting flux moments of the upstream tetrahedron. Once computed, the entering face distribution coefficients are rotated from the face to each of the local uvw systems. The advantage of this is that the coefficients only need be computed (or rootsolved) once for each input face. If the

face is subdivided among sub-tetrahedra, as in cases 2, 3, and 4, the coefficients are transformed into the local coordinate system for each surface.

After the entering face coefficients have been computed, the splitting routine is called and the sub-tetrahedra coordinate systems and rotation matrices are generated. In the local uvw system, the cell and exiting flux moments can be computed using the appropriate spatial quadrature and the equations derived in Chapter 3. Once computed, the moments of the cell flux distribution for each sub-tetrahedron are transformed into the centroid coordinate system where they are combined with the moments from other sub-tetrahedra to form the parent cell flux moments. The exiting face moments for each sub-tetrahedron are transformed into the global face system. For case -2, -3 and case 4 tetrahedra, the exiting face moments for several tetrahedra must be combined into the parent face results.

After the parent tetrahedron results are assembled, the current angle contributions to the scalar flux and currents for the tetrahedron are added to running totals. The exiting face flux moments are used to update the input information for neighbor tetrahedra. The neighbor tetrahedra are then checked to see if they now have the necessary input information and, if so, are appended to the stack and the pointer incremented. This process is repeated until the stack pointer decrements to zero, indicating that the stack is empty, and all spatial cells have been computed for the current angle. A new angle is then chosen and the process repeated until all angles in the quadrature have been used. The current iteration scalar flux is compared to the previous iteration scalar flux for each

tetrahedron to determine if the solution has converged. The comparison for each tetrahedron is performed by computing an error ratio, ϵ_R, defined as

$$\epsilon_R = 2 \frac{| \phi_A^{k-1} - \phi_A^k |}{| \phi_A^{k-1} | + | \phi_A^k |}, \tag{110}$$

where the superscript k indicates the iteration number. If the largest error ratio is smaller than the specified convergence tolerance, the inner iteration is converged, otherwise the current group source is updated using the just-computed scalar fluxes, and the process repeated until the group scalar flux (within group scattering source) converges. When the inner iteration converges, the current group scalar flux is used to update the down scatter source for the lower energy groups. A new inner iteration begins using the next lowest energy group and continues until all of the energy groups have been used. At this point, execution is complete and the output file is generated.

C. Pseudocode Algorithm

The following is a Pseudocode outline of the TetSN algorithm:

Read input data

Initialize angular quadrature

Compute face normals

Initialize scalar flux arrays for current problem

For each energy group:

Do

Save previous iteration scalar fluxes, reset scalar flux arrays

Update the source moments for current group

Generate source coefficients for each tetrahedron if LC, EC, CEC, or CLC

For each angle:

Reset angular flux moment arrays to zero

Compute dot products of face normals and current streaming direction

Determine which tets are ready to compute and append to stack

While stack not empty:

Take tetrahedron from stack, decrement pointer

Determine the case of the current tetrahedron

Split along streaming direction and generate rotation/translation matrices for each sub-tetrahedron

Solve for the coefficients of the entering flux distribution for each input face of the parent tetrahedron

For each sub-tetrahedron:

Rotate the coefficients of the source distribution to the local uvw coordinate system

Rotate the coefficients of the input flux distribution to the local uvw system

Compute the cell flux and exiting flux distribution moments

Rotate\translate the cell flux moments to the centroid coordinate system

Rotate/translate the exiting flux moments to the face system

Next sub-tetrahedron

Assemble parent cell flux moments from sub-tetrahedron moments

Assemble parent exiting face moments from sub-face moments

Update neighbor tetrahedra face flux values and append to stack those now ready to compute

Accumulate scalar flux moments and currents for current tetrahedron

Next tetrahedron from stack

Next Angle

Check for within group convergence by comparing previous iteration scalar flux to current iteration fluxes

Loop until Converged

Compute current group contribution to lower energy group down scatter sources

Next Energy group

Generate output

V. Testing

The characteristic quadratures in the TetSN code were evaluated on four test problems. The first two test problems consist of simple geometries that by no means test the complex three-dimensional solution capabilities, but do demonstrate basic operation and convergence of the methods without unnecessary clutter. The geometry for test problems 3 and 4 is fully three-dimensional. Although all of the methods are tested, heavy emphasis is placed on the first-moment methods, LC and EC. The constant face methods operate correctly, but their low order of convergence and lack of accuracy limits their use in practice, and as a result, they are of academic interest only.

Test problems 1 and 3 are benchmarked against MCNP, (Monte Carlo Neutron Photon Transport Code System) (Briesmeister: 1991). The MCNP code, developed at Los Alamos National Laboratory, is a Monte Carlo simulation code for neutral particle transport in general three-dimensional geometry. It is widely used and accepted in the transport community. The MCNP result is the average of the computed quantity over all histories, $\tilde{x}$, defined as

$$\tilde{x} = \frac{1}{N} \sum_{i=1}^{N} x_i , \tag{111}$$

where N is the number of histories calculated in the problem and x_i is the value of computed quantity for the i^{th} history. MCNP also reports the estimated statistical relative error at the 1σ level, R, defined as $R = S_{\tilde{x}}/\tilde{x}$ where $S_{\tilde{x}}^2$ is the estimated sample variance.

Test problem 1 demonstrates the basic operation of the code, and correctness of the individual methods. The convergence rate, sensitivity to mesh variations, and execution speed of each method are evaluated on test problem 1. The test problem 1 results were benchmarked against MCNP solutions. MCNP was used as an independent check to ensure that there were no gross errors in the derivations and implementations. Test problem 2 is a deep shielding problem used to demonstrate the superior performance of the EC method on very thick cells. The test problem 2 results are benchmarked using a semi-analytic solution that treats the spatial variable exactly but uses a discrete ordinates angular approximation. Test problem 2 has slab symmetry in order to make the semi-analytic solution feasible. Test problem 3 utilizes true three-dimensional geometry to demonstrate the effectiveness of the methods in three dimensions, and to illustrate the need to conserve volumes when meshing regions, a limitation of the current mesh generation method. Test problem 3 results are benchmarked against MCNP solutions. Test problem 4 uses a three-dimensional problem to illustrate the effectiveness of the EC method on thick cells and the ability to use the code to generate flux maps on a problem that would be difficult to solve using Monte Carlo. Because the problem contains curved surfaces, it also demonstrates the power of tetrahedral meshes over traditional boxoid meshes.

5.1. Test Problem 1 - Unit Source Cube with Vacuum Boundaries

The first test problem is a simple 1x1x1 cm cube with vacuum boundaries and a uniformly-embedded isotropically-emitting source of strength 1.0 cm^{-3} sec^{-1}. The cube material has a scattering cross section of 0.5 cm^{-1} and a total cross section of 1.0 cm^{-1}. The meshes for test problem 1 were originally generated using the IDEAS code, but the convergence data generated using them was contaminated by the mesh sensitivity variations to be discussed in Section 5.1.2. To isolate the convergence data from the sensitivity variations, a series of structured meshes was generated manually. To construct each mesh, the problem domain was first meshed using cubic cells, then each cube was subdivided into six tetrahedra. This resulted in a series of structured, self-similar meshes, the first three of which are shown in Figure 6.

Table 1. Mesh Sizes Used for Test Problem 1.

Mesh Number	Number of Tetrahedra
1	6
2	162
3	1296
4	4374
5	10368
6	20250

In analogy to the rectangular case, $1/(n_{tets})^{1/3}$ is an approximation to Δx, the average linear cell dimension. For the test problem 1 mesh sizes, given in Table 1, the

average cell thickness in mean free paths ranges from 0.55 to 0.036. The maximum optical thickness ranges from 1.7 mfp for the coarsest mesh to about 0.1 on the finest mesh. The test problem 1 solutions were computed using the S_{16} level symmetric quadrature and a convergence tolerance of 10^{-6} .

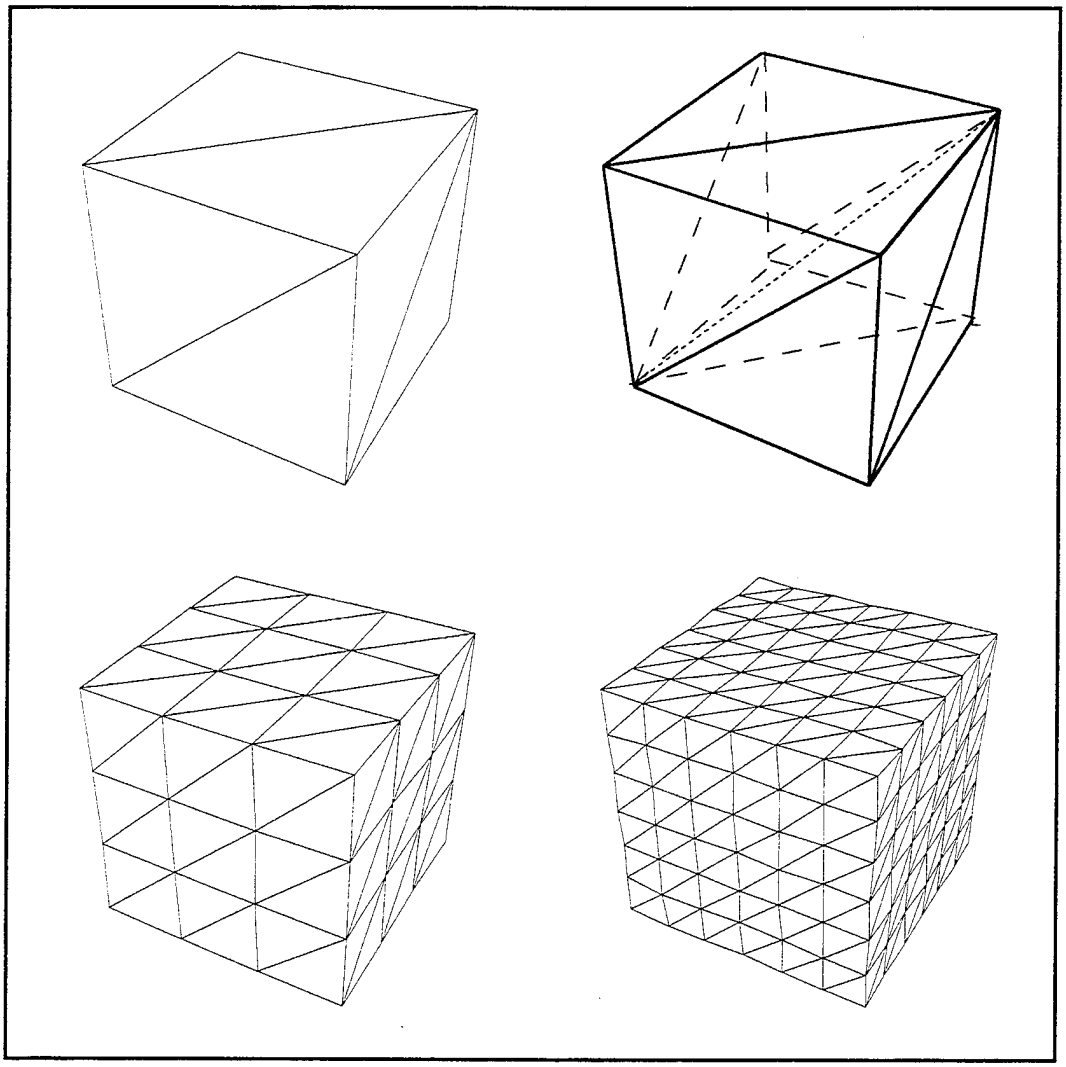

Figure 6. First Three Test Problem 1 Mesh Refinements.

5.1.1. Convergence

There are two levels of convergence to consider in evaluating the performance of the TetSN code. The first is convergence of the TetSN solution to the "correct" solution, i.e. the result that would be obtained in the laboratory. Convergence in this sense is obviously important because to be useful, the discrete ordinates solution must accurately approximate reality. Because the methods use discretized spatial and angular variables, the solutions generated for a given mesh and angular quadrature order are approximate. In theory, as the spatial and angular domains are refined the discrete ordinates solution should converge upon the exact solution.

The second level of convergence is a subset of the first, and illustrates the efficiency of the spatial quadrature used to obtain the discrete ordinates solution. In this case we are concerned with how, for a given angular quadrature order, a particular spatial quadrature approaches the approximate numerical solution. Even if the spatial variable were treated exactly, the solution would only be an approximation due to the angular discretization. As the spatial mesh is refined while the angular quadrature order is held constant, the solutions approach a semi-analytic result in which the angular variable is discretized and the spatial variable is treated exactly.

As the mesh is refined, the average linear cell length, Δx, decreases by a factor $n = \Delta x_1 / \Delta x_2$ where $\Delta x_1 > \Delta x_2$ are the average cell lengths for each refinement. In the thin cell limit, a factor n reduction in the linear dimensions of a spatial cell should result in

a reduction in the solution error given by

$$\frac{\epsilon_1}{\epsilon_2} = n^p \tag{112}$$

where ϵ_1 and ϵ_2 are the solution errors, relative to the benchmark result, for the two meshes, and p is the order of convergence. For methods with a large order of convergence, on fine meshes, further refinement rapidly decreases the truncation error so that greater accuracy is easily achieved.

We first examine the convergence of the spatial quadratures to the fully spatially-converged S_{16} discrete ordinates solution. The problem average flux as computed by TetSN using each of the quadratures is plotted in Figure 7 and Figure 8. The exiting face current results for a single face are shown in Figure 9 and Figure 10. Because of the symmetry of the problem, all faces are equivalent. The first order methods appear to have converged to the same solution on the finest (20250 cell) mesh. The constant face methods are approaching the first order solution, but much more slowly. This is better seen by examining the relative error at each mesh refinement.

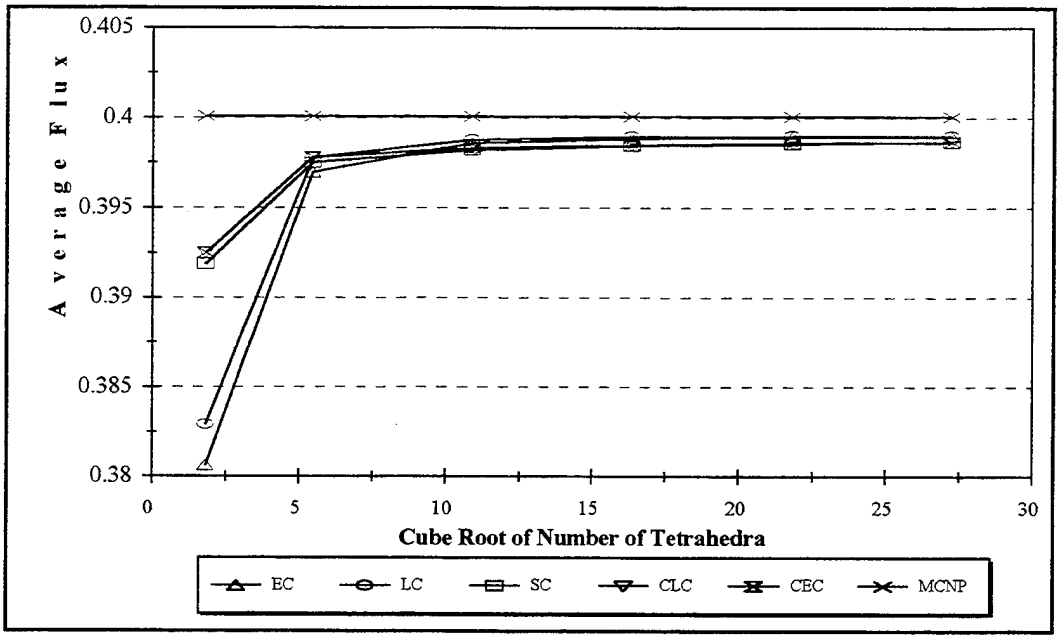

Figure 7. Problem Average Flux for Test Problem 1.

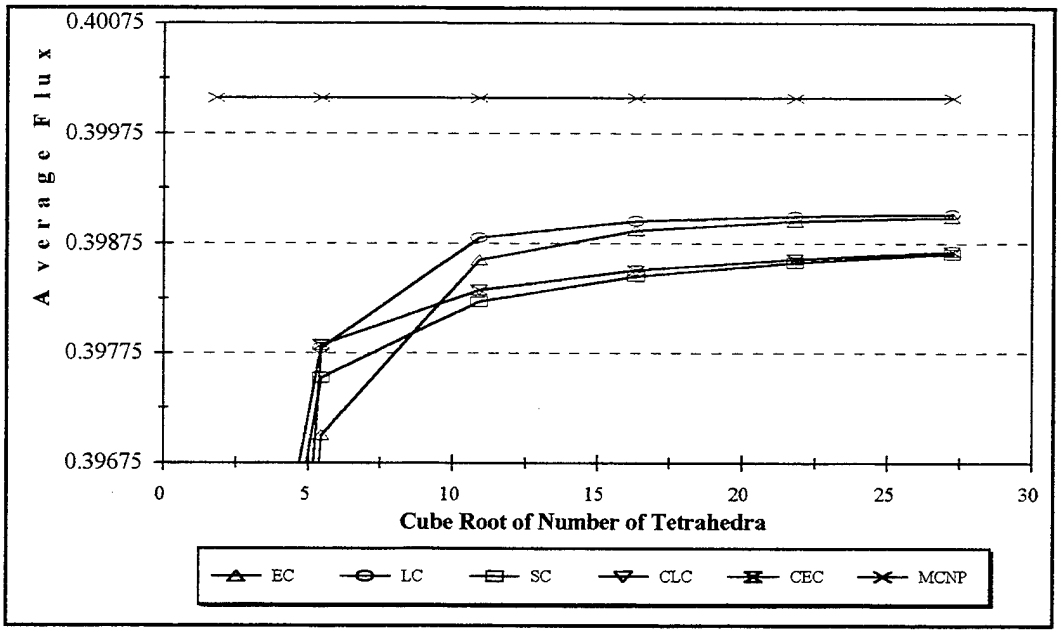

Figure 8. Problem Average Flux for Test Problem 1.
(Enlargement of fine mesh results)

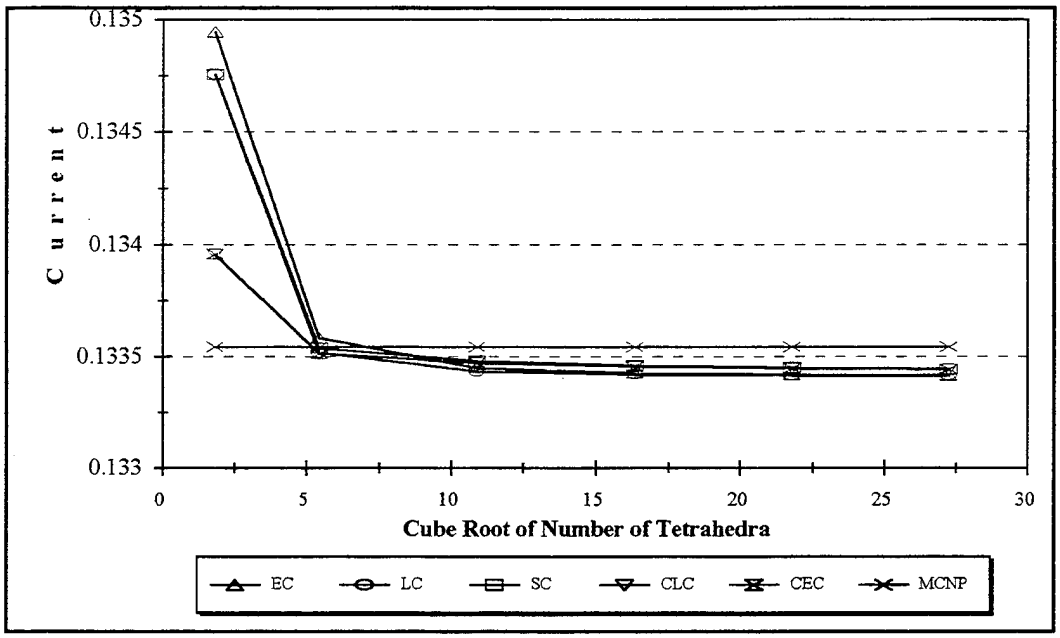

Figure 9. Exiting Face Flux for Test Problem 1.

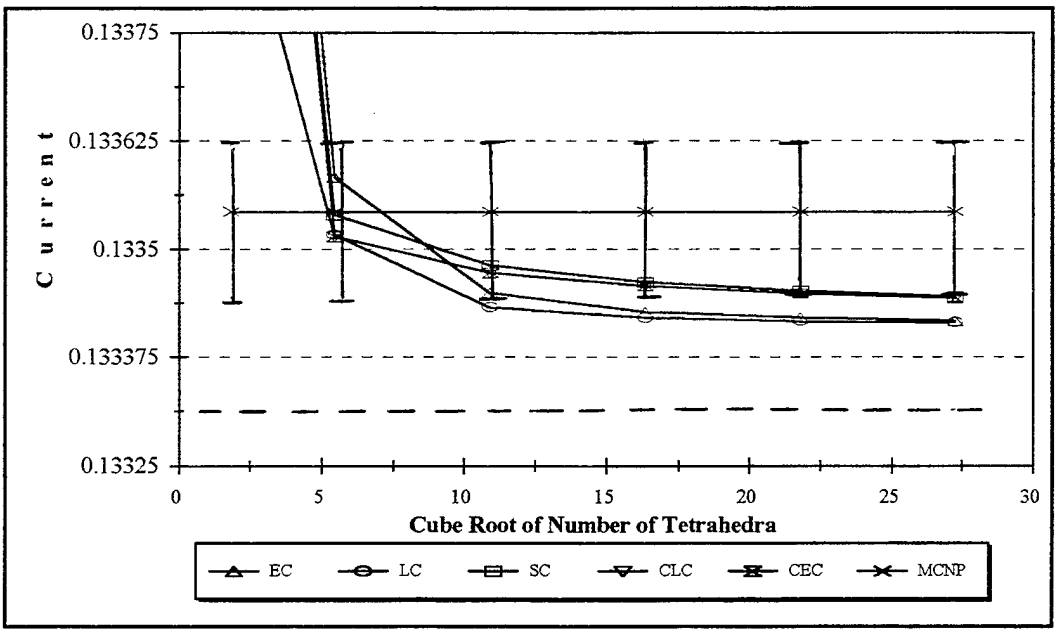

Figure 10. Exiting Face Current for Test Problem 1.
(Enlargement of Fine Mesh Results)

In Figure 11 and Figure 12, the absolute relative error, ϵ_{ar}, of the solution for each spatial refinement with respect to the benchmark discrete ordinates solution is plotted against the cube root of the number of tetrahedra in the mesh for each of the methods. The absolute relative error between a given value, ϕ, and a benchmark result, ϕ^{bench}, is computed using

$$\epsilon_{ar} = \frac{|\phi^{bench} - \phi|}{\phi^{bench}} . \tag{113}$$

Because we are concerned with the convergence of the spatial quadrature to the S_{16} discrete ordinates solution as the mesh is refined, we use the most converged TetSN result, the 20250 tetrahedron mesh LC solution, as the benchmark. On the log-log plot, the order of convergence for a method is simply the slope of its corresponding curve. EC and LC both exhibit approximately third order convergence. This is in agreement with the third-order convergence rates observed for these methods in two-dimensional rectangular and triangular schemes (Minor, 1993: 91-94; and Miller, 1993:74-76). The constant face methods however, exhibit slightly less ($p \sim 0.9$) than the first-order convergence rates observed for the SC method (Minor, 1993: 91-94; and Mathews, Miller and Brennan, 1996). As discussed in Section 2.2 , using constants to represent the entering flux distribution can cause numerical diffusion. The constant face quadratures, as derived for tetrahedral cells, appear to be more sensitive to the treatment of the entering face flux (see

also Section 5.1.2) than is the case in one or two dimensions, contributing to their slow convergence and lack of accuracy.

It should also be noted that the MCNP results for the problem average flux and exiting current are inconsistent. The MCNP result gives a higher problem average flux than the TetSN solutions. This being the case, we expect the MCNP exiting current to be lower. This is not the case. Because of the nature of the estimators used in the Monte Carlo calculation, particle balance is not strictly satisfied. If we recompute the value of the exiting current based on the MCNP average flux result and conservation of particles, we get the result shown in the dashed line of Figure 9 and Figure 10. Indeed, the TetsN solutions for the exiting current are converging toward the particle conserving MCNP results.

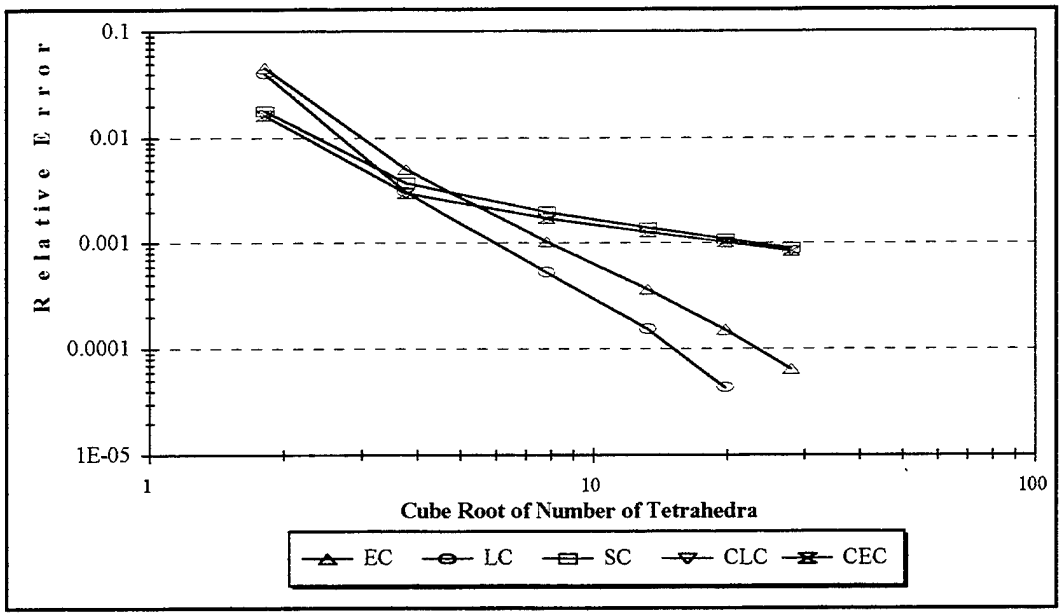

Figure 11. Relative Error in Test Problem 1 Problem Average Flux.

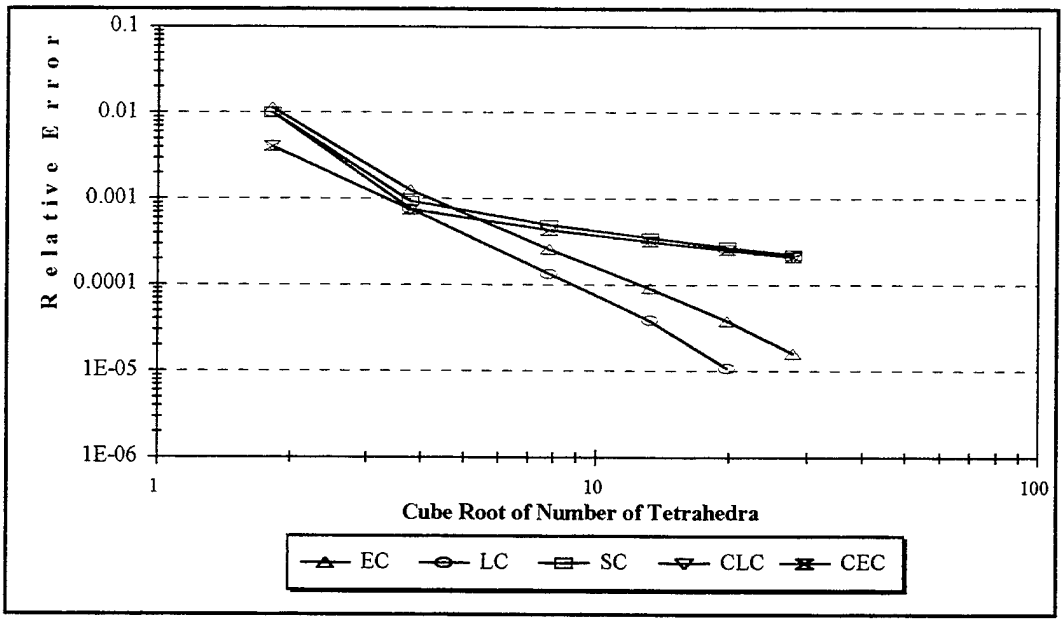

Figure 12. Relative Error in Test Problem 1 Exiting Face Current.

Error norms of the face flux distribution were computed to estimate the pointwise convergence properties of the quadratures. The error norms were computed by mapping a regular mesh of points onto the surface. For each point, the flux value was computed based on the computed flux representation for the tetrahedron face containing the point. This allowed us to compare the distributions of two separate meshes having dissimilar face arrangements.

Two norms were computed: L_∞ and L_1. The L_∞ norm is defined as the maximum of the absolute value of the relative errors over the face,

$$L_\infty = \max_{n=1,N} \frac{|\phi(x_n,y_n) - \phi^{bench}(x_n,y_n)|}{\phi^{bench}(x_n,y_n)}, \tag{114}$$

where N is the number of points used, $\phi(x, y)$ is the flux distribution at point (x, y) for the current mesh refinement, and $\phi^{bench}(x, y)$ is the flux at point (x, y) for the benchmark solution. The L_1 norm is defined as the average of the absolute relative errors over the face,

$$L_1 = \frac{1}{N} \sum_{n=1}^{N} \frac{|\phi(x_n, y_n) - \phi^{bench}(x_n, y_n)|}{\phi^{bench}(x_n, y_n)} . \tag{115}$$

The norm computations were repeated using successively finer point spacings until the L_1 norm converged to three digits and the L_∞ norm converged to two digits (the L_∞ norm is more sensitive to small changes in the point spacing). This required using a 50x50 mesh of points.

Figure 13 shows the L_∞ norm for the exiting face flux distribution compared to the benchmark (20250 tetrahedron LC result) at each spatial refinement for each of the quadratures. The average relative error of the exiting flux is shown in Figure 14. The first order method norms show approximately second order convergence. Although the constant face methods do a reasonable job of approximating the average value of the exiting flux, we should not expect them to accurately approximate the pointwise distribution. The constant solution on a face agrees with the pointwise values only where the pointwise solution equals the average. Even for a relatively flat distribution, the constant face methods require a fine mesh to adequately resolve the face flux spatial distribution.

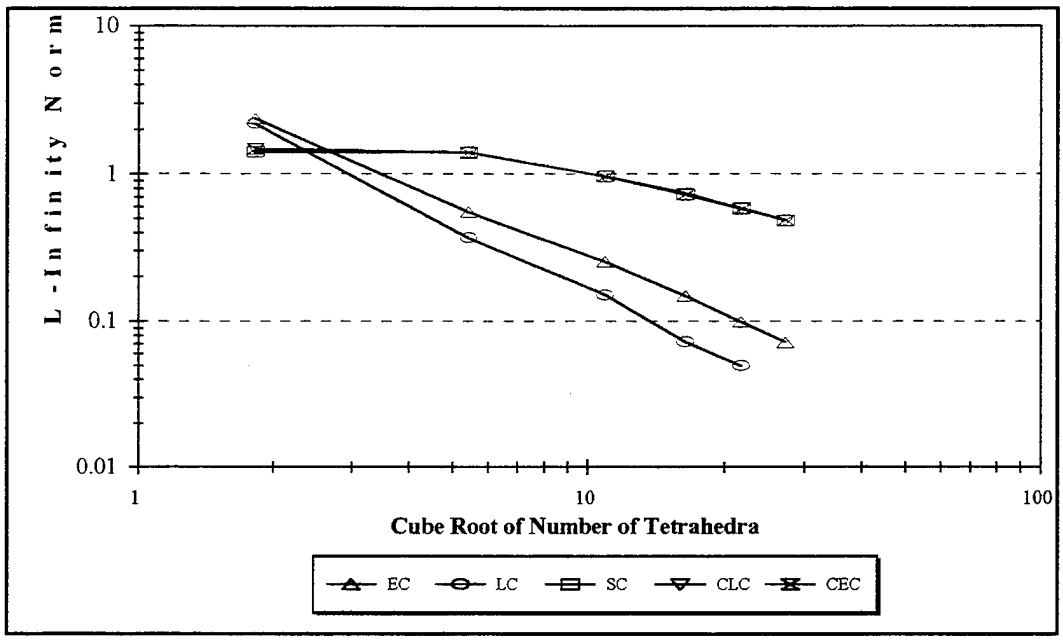

Figure 13. L-Infinity Norm (Maximum Relative Error) of Test Problem 1 Exiting Face Flux Distribution.

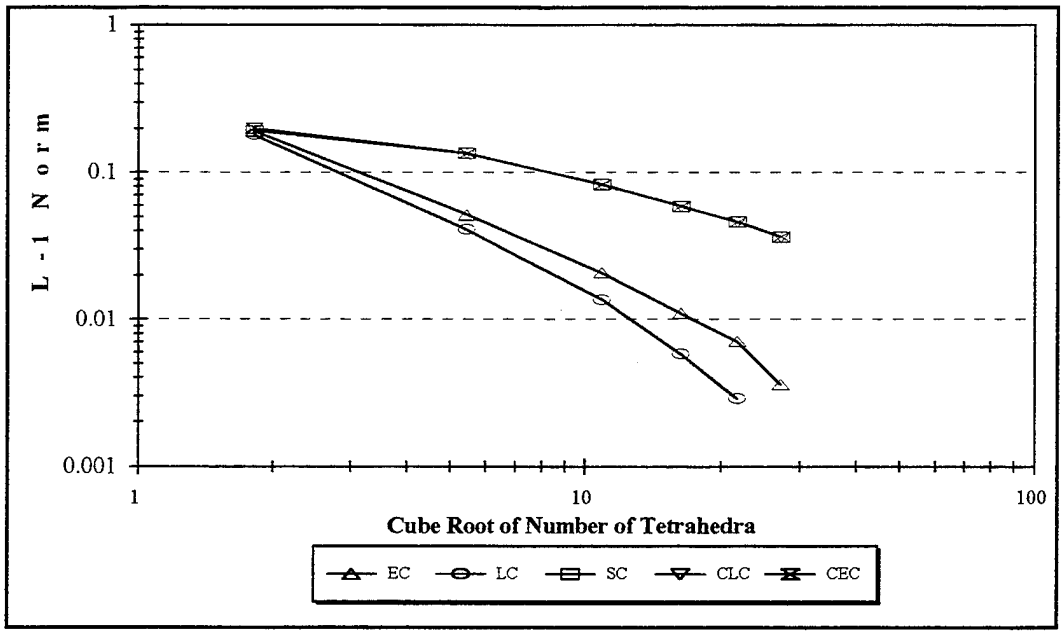

Figure 14. L-1 Norm (Average Relative Error) of Test Problem 1 Exiting Face Flux Distribution.

The Monte Carlo solution method uses continuous angular and spatial variables and contains only statistical errors. Because of the relative simplicity of the problem geometry and cross sections, the MCNP solution represents, to within statistical uncertainty, the laboratory solution. The MCNP solution is also plotted in Figure 7 through Figure 10. The uncertainty in the MCNP solution for the average flux is 0.02%, which is too small to represent with error bars in Figure 7, but is shown in Figure 8. The MCNP exiting current uncertainty was 0.08%. The difference in the fully converged TetSN and MCNP solutions is a result of the discretized angular treatment used in the discrete ordinates calculation. Overall, we expect that as the solution approaches the continuous solution, i.e. as the discretized spatial and angular variable mesh spacing approaches zero, the discrete ordinates solution should approach the actual (in this case MCNP) solution. Figure 15 shows the relative error between TetSN EC solutions for various mesh refinements and angular quadrature orders and the MCNP solution. Figure 16 and Figure 17 show the same for the LC and SC methods respectively. The figures show that each method converges to the MCNP solution as either the angular quadrature order or number of tetrahedra in the mesh is increased, as it should for a properly running code.

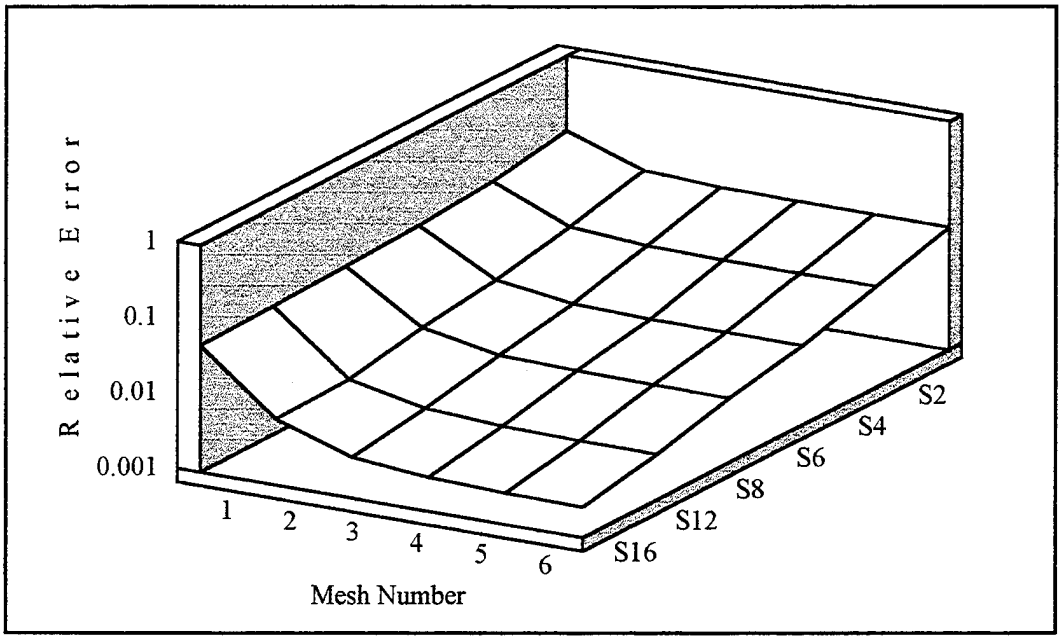

Figure 15. Relative Error Between TetSN EC and MCNP Solutions for Test Problem 1 Problem Average Flux.

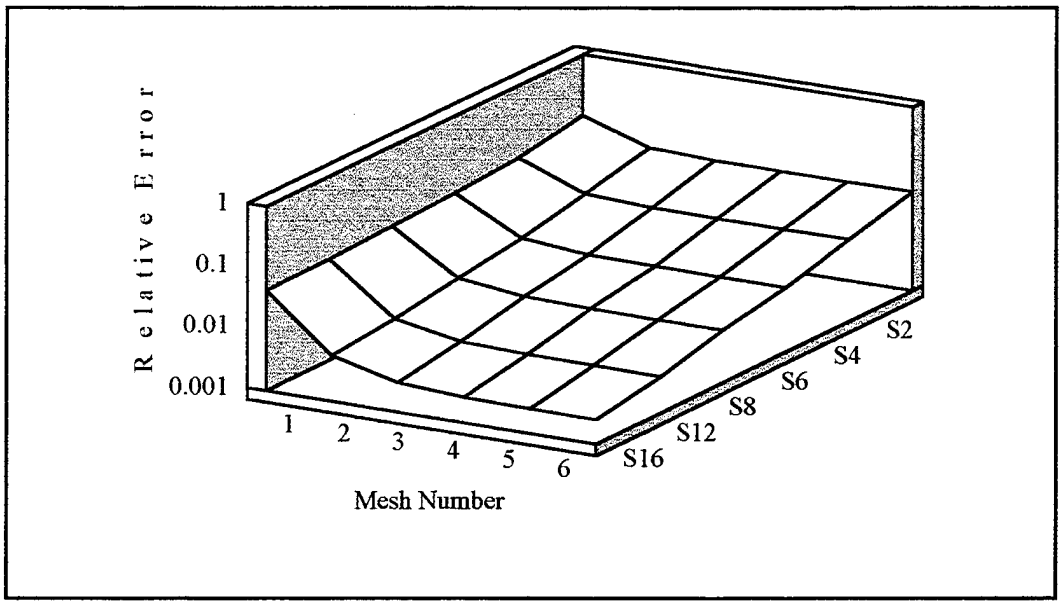

Figure 16. Relative Error Between TetSN LC and MCNP Solutions for Test Problem 1 Problem Average Flux.

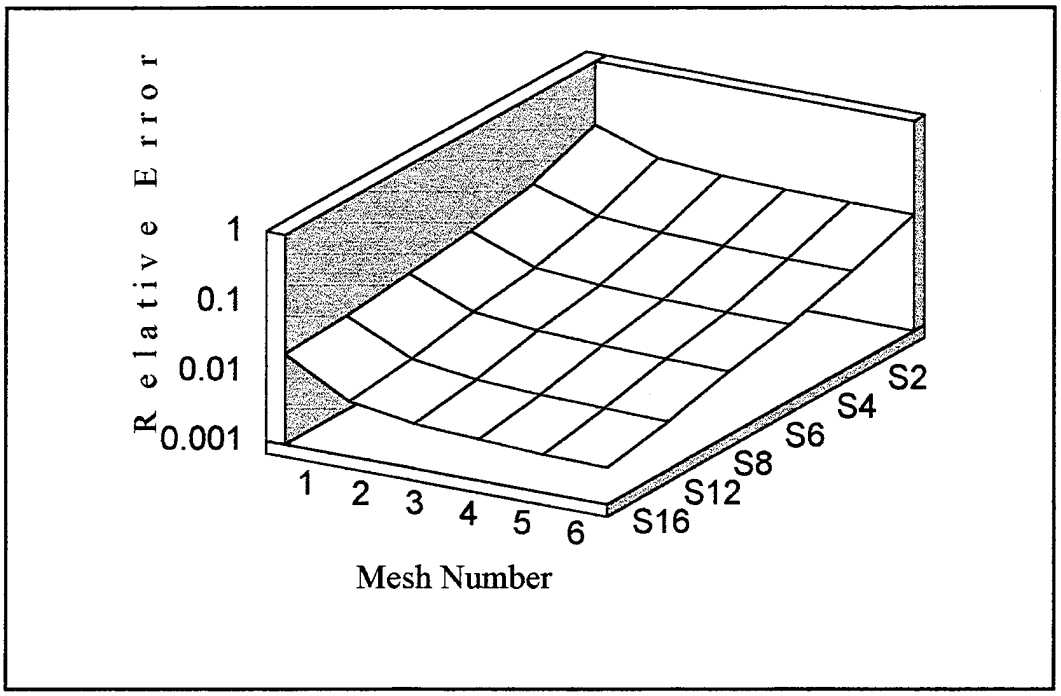

Figure 17. Relative Error Between TetSN SC and MCNP Solutions for Test Problem 1 Problem Average Flux.

5.1.2. Mesh Sensitivity

A robust method should produce consistent results on a mesh containing a given number of cells, regardless of the particular arrangement of the individual cells. If this is not the case, then it is impossible to have any confidence in any one solution obtained on an arbitrarily generated mesh. To study the effects of mesh variations, each method was used to generate solutions for test problem 1 on a series of ten different meshes, each containing 162 cells. The modified meshes were generated by randomly varying the x, y,

and z coordinates of each node in the original mesh (corner nodes were fixed and the face and edge nodes were constrained to variations in the face plane and edge line respectively so as to retain the original problem boundaries). The node coordinates were allowed to vary by up to 25% or 50% of the distance between the two closest nodes in the mesh, i.e.

$$\begin{aligned} x_{new} &= \epsilon_{var}\, d_{\min}\,(2\,\nu - 1) + x_{orig} \\ y_{new} &= \epsilon_{var}\, d_{\min}\,(2\,\nu - 1) + y_{orig} \\ z_{new} &= \epsilon_{var}\, d_{\min}\,(2\,\nu - 1) + z_{orig}\,, \end{aligned} \tag{116}$$

where d_{min} is the minimum mesh point spacing, ϵ_{var} is the maximum fractional variation allowed (0.25 or 0.5), ν is a random number that varies between 0 and 1, and x_{orig}, y_{orig}, and z_{orig} are the original node coordinates. This produced the two sets of meshes shown in Figure 18 and Figure 19.

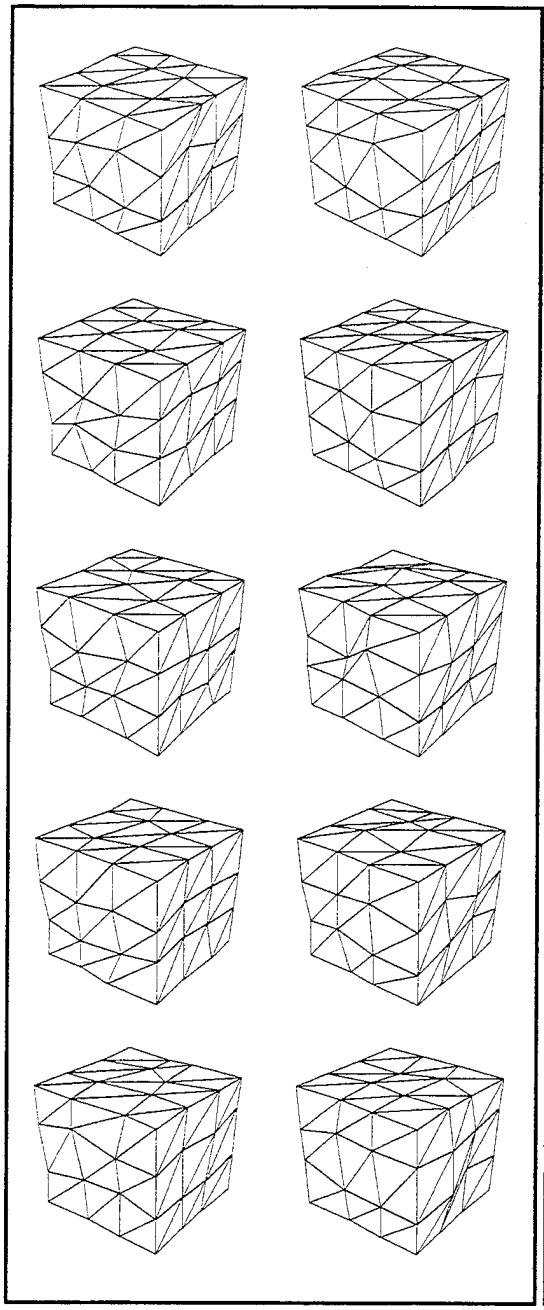

Figure 18. Exterior View of 25% Variation Meshes. (Apparant distortion of the cube is artifact of file transfer)

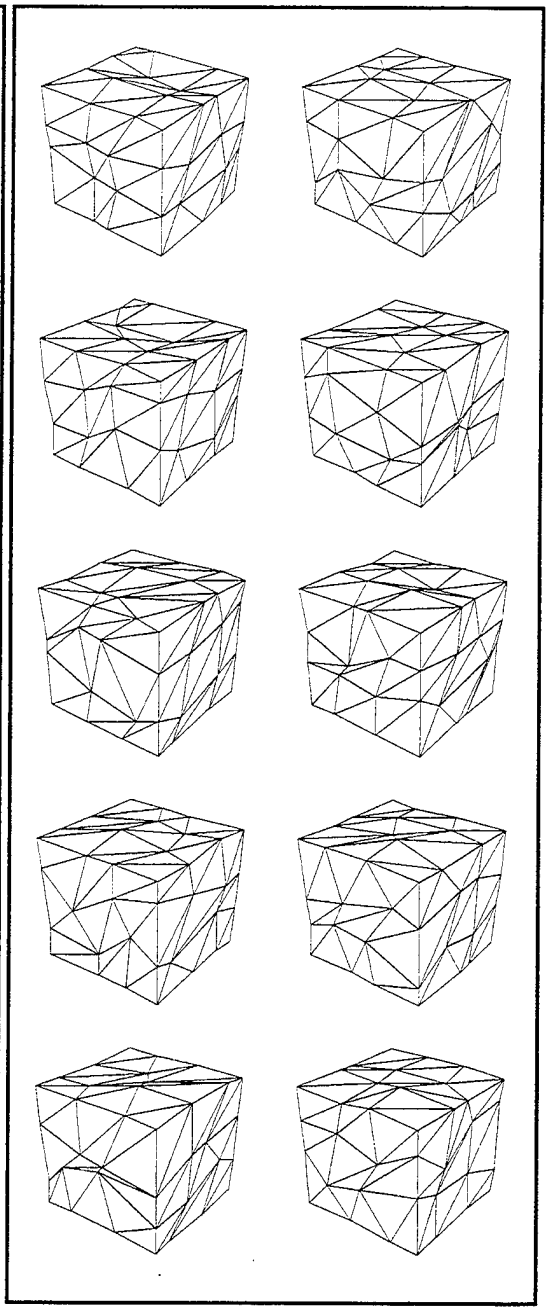

Figure 19. Exterior View of 50% Variation Meshes.

Before examining the influence of the mesh variations, we examine the characteristics of the meshes themselves. As a quantifiable measure of tetrahedron shape, we define the aspect ratio of a given tetrahedron to be the ratio of the tetrahedron volume to that of an equilateral tetrahedron in which the edges are the length of the longest edge of the tetrahedron in question. For this metric, an ideal (equilateral) tetrahedron has a volume ratio of one, and a poorly shaped tetrahedron has a volume ratio approaching zero. The cumulative distribution functions (CDF) of volume ratios for the two sets of meshes are shown in Figure 20 and Figure 21. The Heaviside distribution in the figures is the structured mesh CDF, for which all tetrahedra have a volume ratio equal to 0.27.

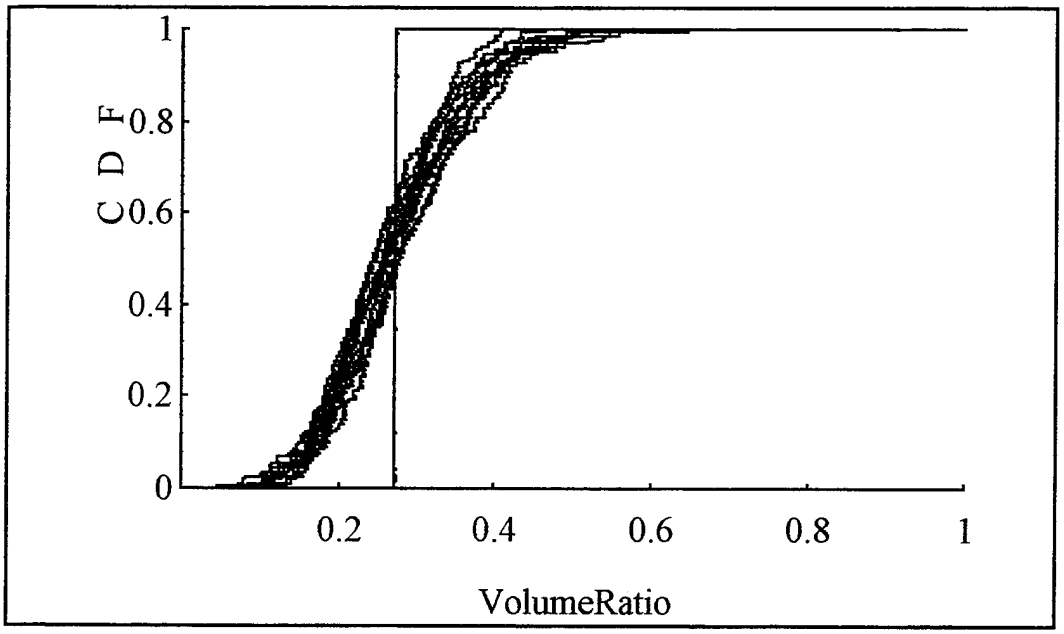

Figure 20. Cumulative Distribution Function of Tetrahedron Volume Ratios for the 25% Variation Meshes.

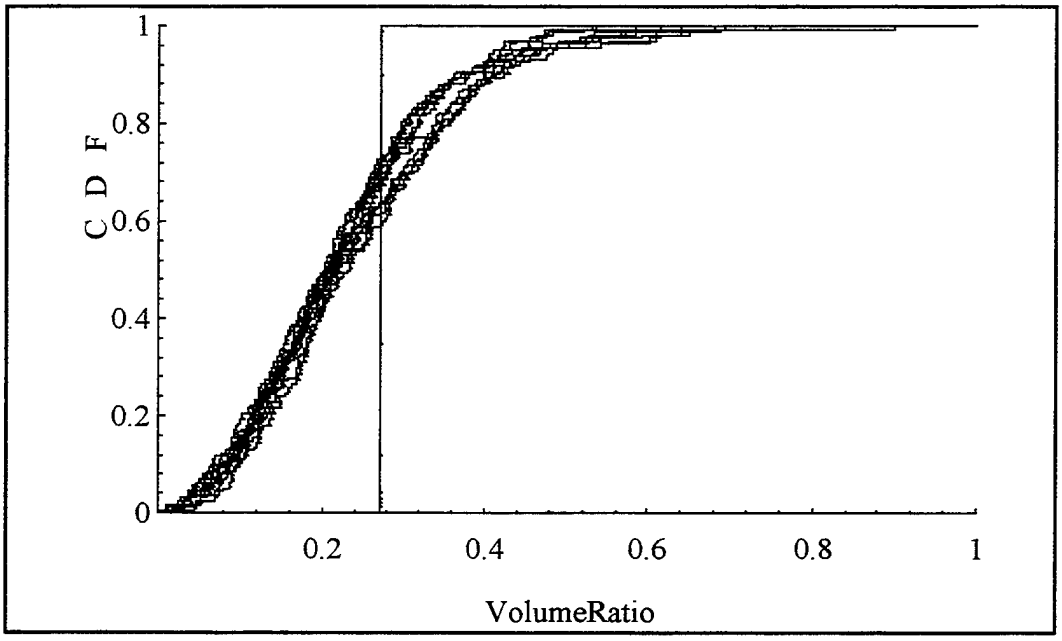

Figure 21. Cumulative Distribution Function of Tetrahedron Volume Ratios for the 50% Variation Meshes.

The effect of the variations was to produce tetrahedra approximately equally distributed about the structured mesh value. The 50% cases contain more extreme variations, including some very poorly shaped tetrahedra. For the ten 50% meshes, the minimum volume ratios ranged from 0.00017 to 0.019, and the maximum ranged from 0.53 to 0.9. Figure 22 shows the structured and two 50% variation cases (mesh #4 and mesh #5) along with the worst and best shaped tetrahedron from each. The tetrahedron shapes ranged from essentially plane figures to almost perfect equilateral tetrahedra.

To be robust, the code must be able to handle wide variations in the cell aspect ratio. One would hope that a mesh-generating algorithm would at least avoid such dramatically bad shapes. This is indeed the case for the IDEAS generated meshes. A

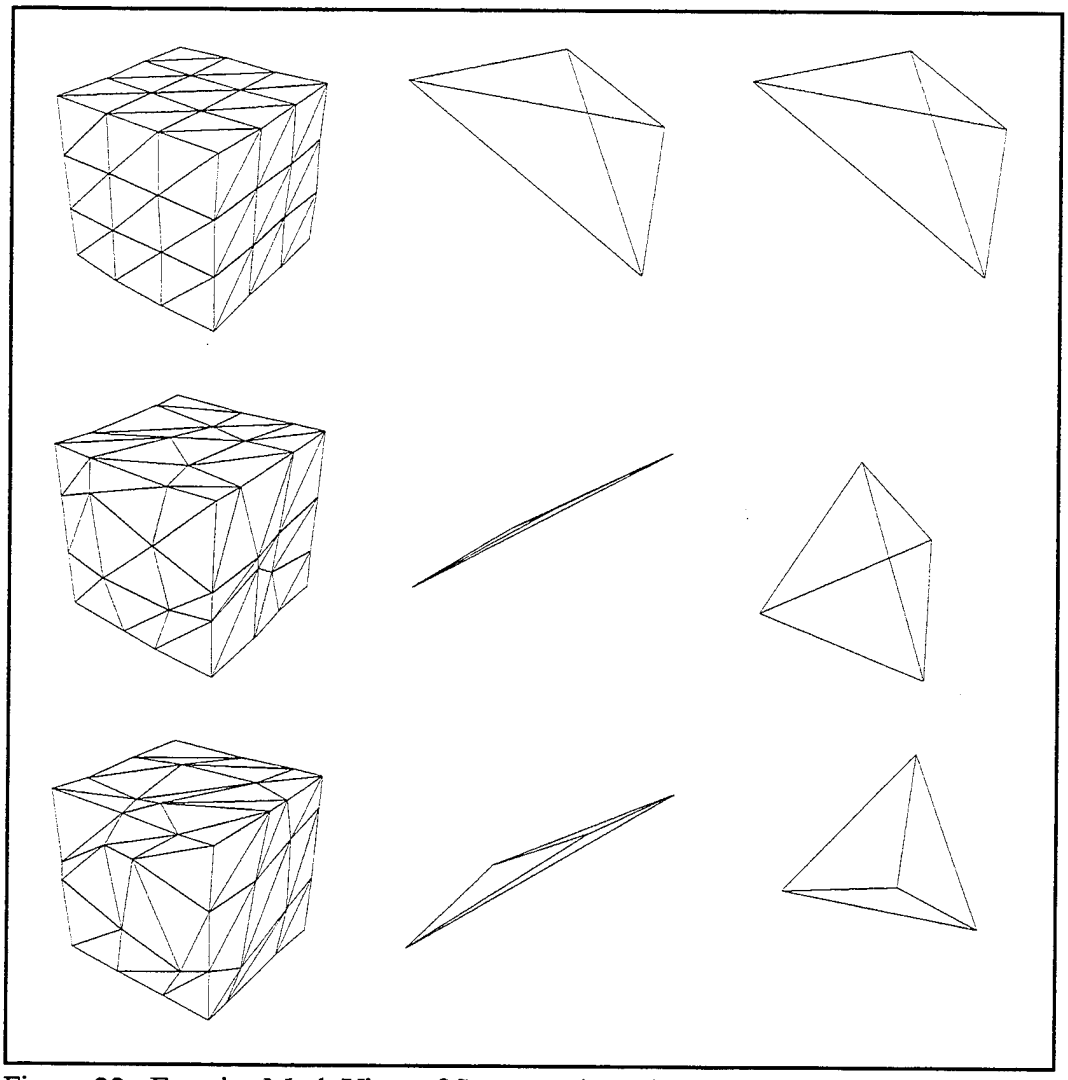

Figure 22. Exterior Mesh View of Structured Mesh, Mesh #4 and Mesh #5 with Worst and Best Shaped Tetrahedron for Each.

CDF for a 482 cell mesh generated by IDEAS is plotted along with the structured mesh CDF and a representative sample from each of the 25 and 50% variations in Figure 22. The tetrahedra generated by IDEAS have, on average, better aspect ratios than the tetrahedra generated by randomization. Hence for most problems, IDEAS should be able to produce acceptably shaped tetrahedra and there is no need to be concerned with

method performance on poorly shaped cells. There are, however, problems for which the shapes and relative sizes of the individual regions in a problem require either many high aspect ratio cells, or a few low aspect ratio cells, to model. For these problems,

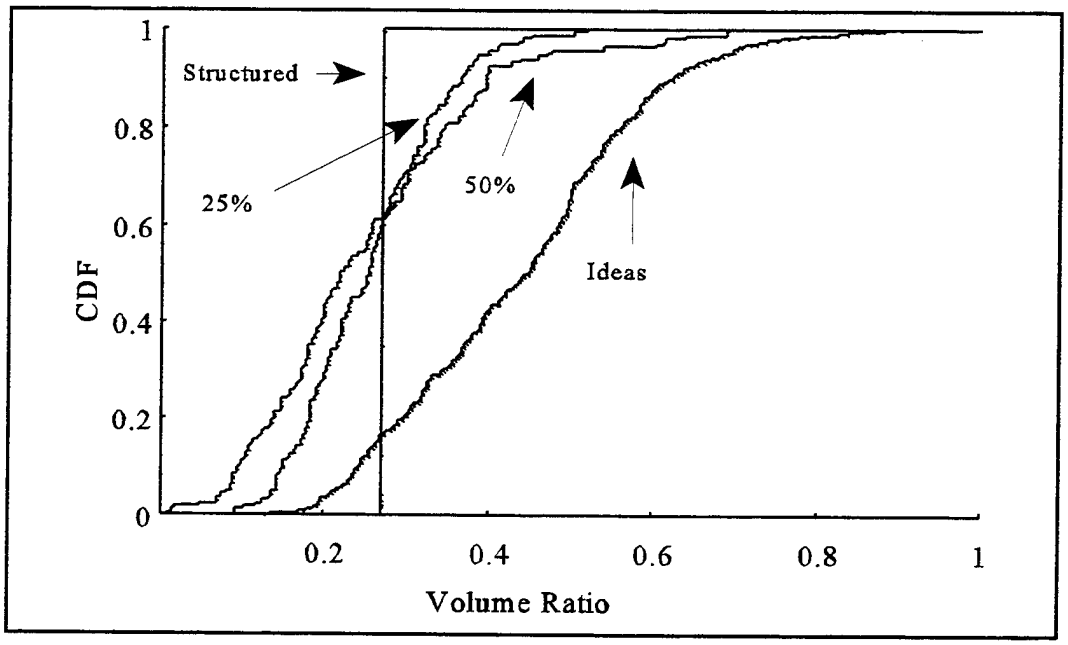

Figure 23. CDF for IDEAS Generated Mesh.

algorithms capable of producing accurate results on poorly shaped tetrahedra offer increased computational efficiency.

Figure 24 and Figure 25 show the relative error in the test problem 1 problem average flux for each of the 50% mesh variations. Variation 0 is the structured mesh. The same is shown for the face flux error norms in Figure 26 and Figure 27. The 25% variations for all computed quantities were smaller than the 50% values, so we examine

only the 50% variation meshes. While the constant face methods show some variations, the relative errors are all of the same order of magnitude or less than the structured mesh solution. The first moment methods were less sensitive to variations in the mesh, with LC performing slightly better than EC. The first moment method solutions for the average flux were accurate to three digits, but the variations produced by the different meshes caused the solutions to change in the fourth digit, hence the effects of the variations for the LC and EC quadratures were entirely insignificant. Note that the presence of the very poorly shaped tets in meshes 4 and 5 caused no significant deviations in the relative errors for those meshes.

As explained in section 2.2, using constants to represent the face flux distributions can lead to truncation errors, namely numerical diffusion. The way the errors generated as a result of numerical diffusion propagate through the mesh depends on the specific orientation of the faces, and hence the cells, in the mesh. This means that the constant face method errors (more so than the first moment method errors), relative to the most converged discrete ordinates solution, depend upon both the number of cells and the specific orientation of the cells. This heightened sensitivity to variations in the mesh composition (Figure 24) required the use of structured meshes to evaluate quadrature convergence in Section 5.1.1.

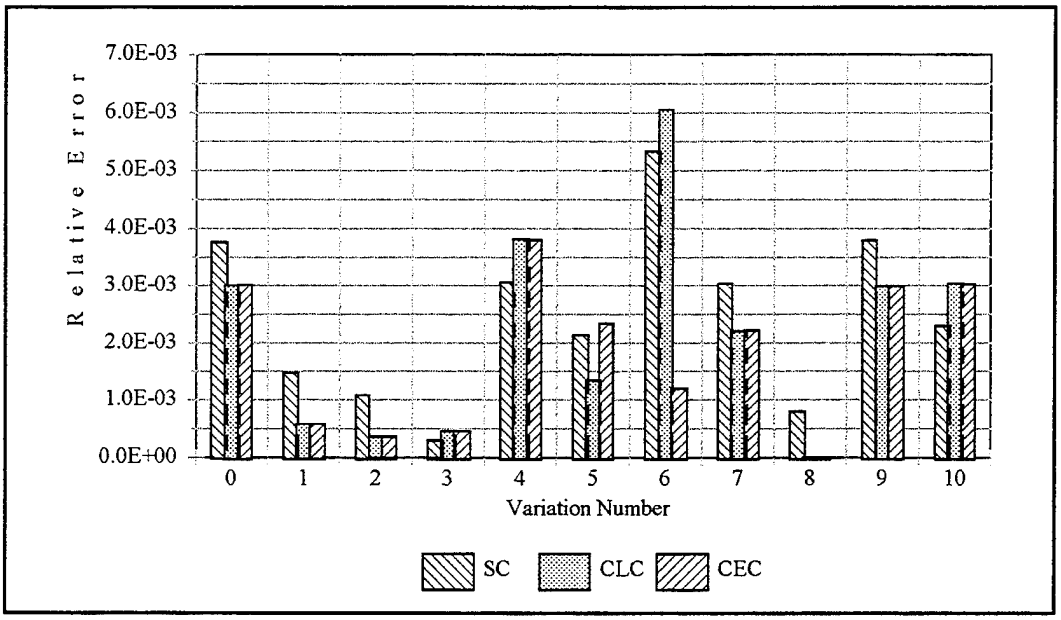

Figure 24. Relative Error in Problem Average Flux for 50% Node Variation.

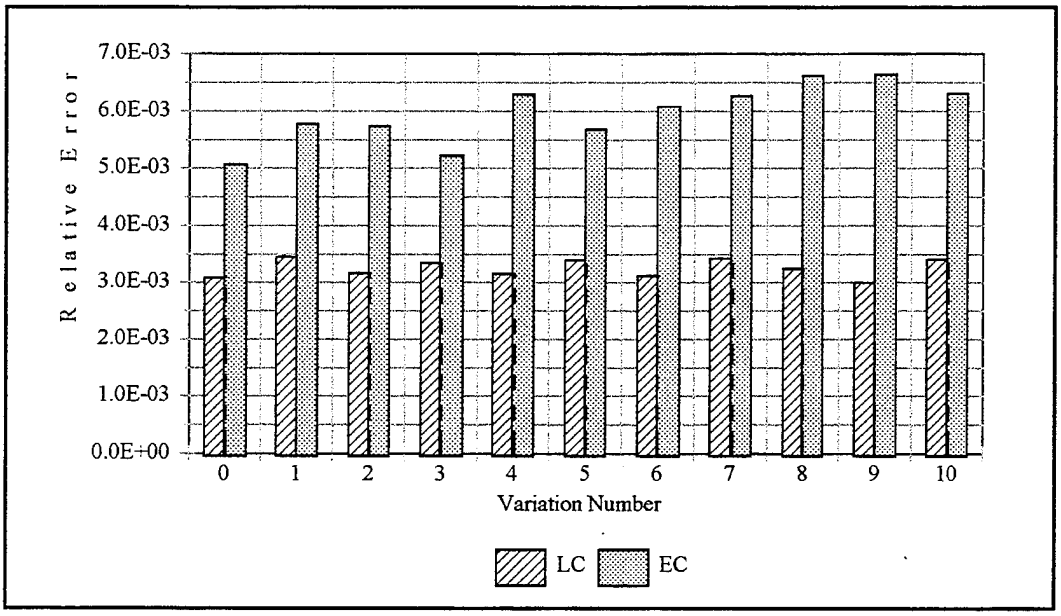

Figure 25. Relative Error in Problem Average Flux for 50% Node Variation.

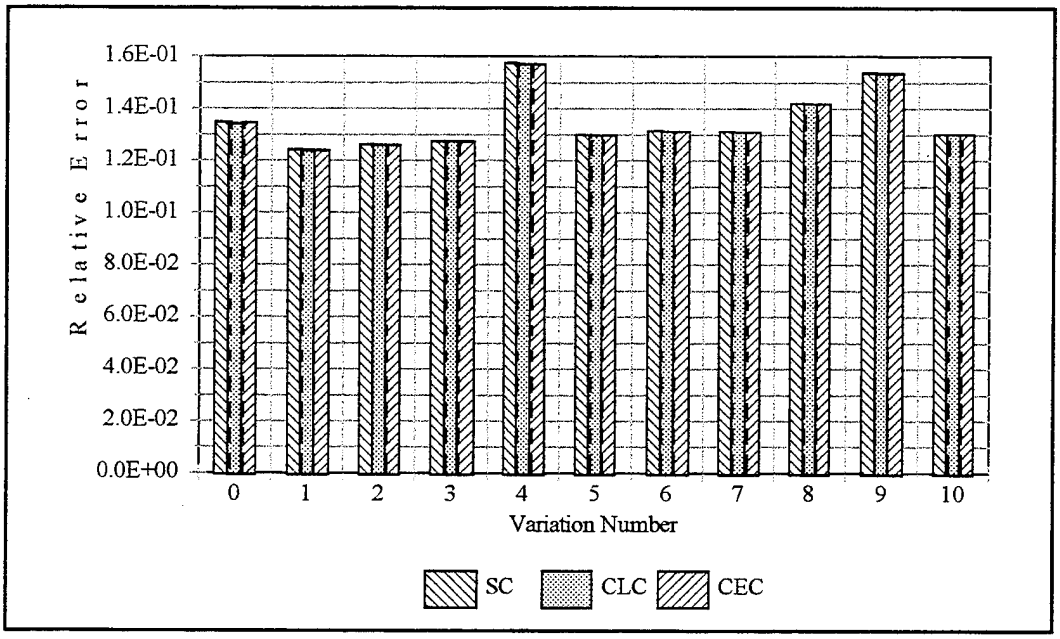

Figure 26. L_1 Norm of Exiting Face Flux for 50% Variation.

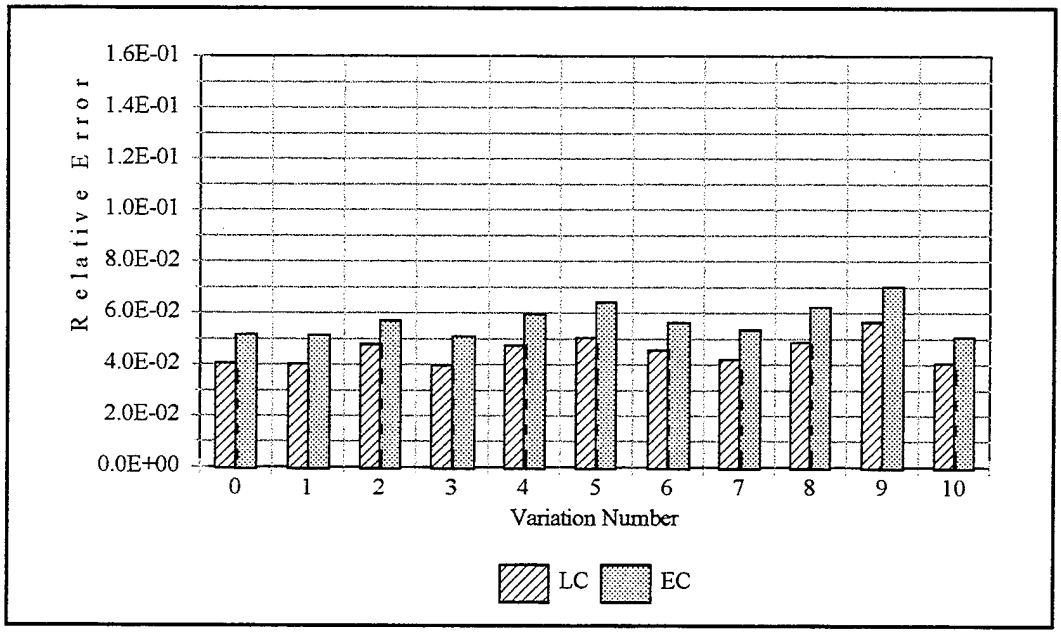

Figure 27. L1 Norm of Exiting Face Flux for 50% Variation.

5.1.3. Execution Speed

The execution times per phase space cell (i.e. per tetrahedron, per angle, per energy group, per iteration) for test problem 1 on an IBM RSC-6000, model 590 computer are given in Table 2.

Table 2. Execution Time per Phase Space Cell (seconds).

	Total	Quadrature /Cnsrv.	Split/ Assemble	Source Moments	Input Moments	Over-head[a]
SC	1.2×10^{-4}	6.4×10^{-5}	3.5×10^{-5}	-	-	2.1×10^{-5}
CLC	1.4×10^{-4}	7.3×10^{-5}	4.2×10^{-5}	1.7×10^{-7}	-	2.5×10^{-5}
CEC[b]	1.1×10^{-3}	9.7×10^{-4}	5.0×10^{-5}	2.9×10^{-5}	-	2.9×10^{-5}
LC	2.0×10^{-4}	7.7×10^{-5}	7.3×10^{-5}	1.7×10^{-7}	4.9×10^{-6}	3.5×10^{-5}
EC[c]	1.6×10^{-3}	1.0×10^{-3}	8.9×10^{-5}	2.9×10^{-5}	3.6×10^{-4}	4.0×10^{-5}

a. overhead includes reading input; generating neighbor, face, and centroid lists; accumulating scalar fluxes; computing dot products; resetting arrays between iterations; updating neighbor triangles; and writing the output file.
b. the CEC method was never modified to use the balance equations to compute the cell source moments.
c. EC (and to a smaller extent CEC) execution times may vary by about ±30% from the listed times depending on the details of the specific problem being solved (see text)

The overhead tasks can be divided into four categories: those required to be performed once per program execution, those required once per iteration, those required once per angle per iteration, and those required once per cell per angle per iteration. The overhead tasks required once per cell per angle per iteration account for about 86% of the overhead time in the problem used to generate the table (162 tets, 9 iterations, 80 angles).

Further, the majority of the remaining tasks scale with the number of spatial cells. Hence, the time required to execute the overhead tasks scales with the number of cells , and the times in the table can be considered to be independent of problem size.

The EC method is the most costly, requiring about 8 times the effort of LC and 14 times the effort of SC per phase space cell (per tetrahedron per angle per iteration). The main difference between the exponential methods and the other methods is that the exponentials in the assumed form of the flux and source distributions produce multi-dimensional moment functions in the resulting equations. The multi-dimensional functions are generally more expensive to compute. The SC, CLC, and LC quadrature moment equations all contain moments functions of a single argument, in fact, always the same argument, ϵ, making them much more efficient. The EC method spends 63% of its execution time computing moments functions, while the LC, CLC, and SC methods spend between 20 and 30%. Conversely, the non-exponential quadratures spend 53% of their execution time splitting and assembling tetrahedra while the EC method spends about 10%. Enhancing the performance of the moments function and splitting/assembly routines represents the largest potential gain in terms of increased computational efficiency

The execution time for the SC, CLC, and LC methods is fairly constant. The execution speed of the CEC and in particular the EC method is subject to variations based on several factors. First, as detailed in Appendix A, the moment function routines use either recursion, function expansion approximation, or a combination of both to compute multi-argument moment functions. The expansions execute faster than recursion, so the

execution time of a multi-dimension moment function will vary based on its arguments. Second, as explained in Appendix B, the root-solving routines can use two separate formulations, based on the values of the arguments, to determine the expansion coefficients. The single-argument formulation for the two- and three-dimensional root solvers is more efficient than the multi-dimensional form, affecting the problem run time. Finally, the EC method uses either Equations (88) and (90) or Equations (108) to compute the cell flux moments. Using the balance equations is more efficient than Equations (88) and (90), causing variations in the execution speed of the EC method.

5.2. Test Problem 2 - Cube Problem with Isotropic Incident Current and Slab Symmetry

Test problem 2 consists of a homogeneous unit cube with reflective boundaries at $x = \pm 0.5$ cm and $y = \pm 0.5$ cm, and vacuum boundaries at $z = \pm 0.5$.cm An isotropic current, $J_z^+(x,y,-0.5\,\text{cm}) = 1.0\ \text{cm}^{-2}\,\text{sec}^{-1}$, is incident on the z = -0.5 cm face. The flux produced by the isotropic entering current is a function of the streaming angle, $\hat{\Omega}_n$:

$$\psi_n(x,y,-0.5\,\text{cm}) = \frac{2J_z^+(x,y,-0.5\ \text{cm})}{\xi_n}, \quad (\xi_n > 0)\,. \tag{117}$$

An isotropic incident current would be produced by an isotropically emitting surface-deposited source. This boundary condition is used since it is more stressing than a Lambertian boundary condition. The Lambertian incident current consists of a constant entering angular flux, i.e. ψ_n = constant, which can give unexpectedly good results that are not a true measure of the quadrature performance. The cube material has a scattering cross section of 4 cm^{-1} and a total cross section of 32 cm^{-1}. The meshes for this problem were all generated using IDEAS and contained from 6 to 22028 cells. The average linear cell thickness for this problem is 18 mfp on the coarsest mesh and 1.1 mfp on the finest mesh. The maximum linear cell dimension is 54 mfp for the coarsest and 3 mfp for the finest mesh. The TetSN solutions for this problem were generated using an S8 level symmetric quadrature and a convergence tolerance of 10^{-6}.

Because the problem is symmetric about two axes, we can compare the TetSN results to a one-dimensional semi-analytic solution with equivalent boundary conditions

(Davison, 1958: 174-175). The semi-analytic results were computed in Mathematica (Wolfram, 1992) using a modified three-dimensional S_8 level-symmetric quadrature. The modified three-dimensional quadrature used only the ξ values (direction cosines with respect to the z-axis) to specify the angles. The weights for this modified quadrature were just the sum of the point weights of all the angles having the same ξ value (the level weights).

Plots of the problem average flux and relative error of the average flux compared to the semi-analytic solution are shown in Figure 28 and Figure 29. All methods are able to compute the average scalar flux to within a few percent even on the coarsest mesh, with LC and in particular, EC, doing the best job. Because the shield is optically very thick, the flux is highest near the entering face and decreases rapidly in the +z direction. There is a thin boundary layer near the entering face, where the entering particles are scattered back out of the shield. The problem average flux is dominated by the flux in this region. The constant face and LC methods are capable of adequately representing the face and cell distributions in this region and hence do a reasonable job computing the problem average flux. Beyond the boundary layer, the flux decays essentially exponentially. The EC method is able to almost exactly model the flux falloff, and computes the problem average flux to within about 1% and exiting face current to within 10% using only six cells. The CEC method, because it is better able to model the scattering source distribution, performs slightly better than the other constant face methods on the coarser meshes, but the overall performance of these methods is poor.

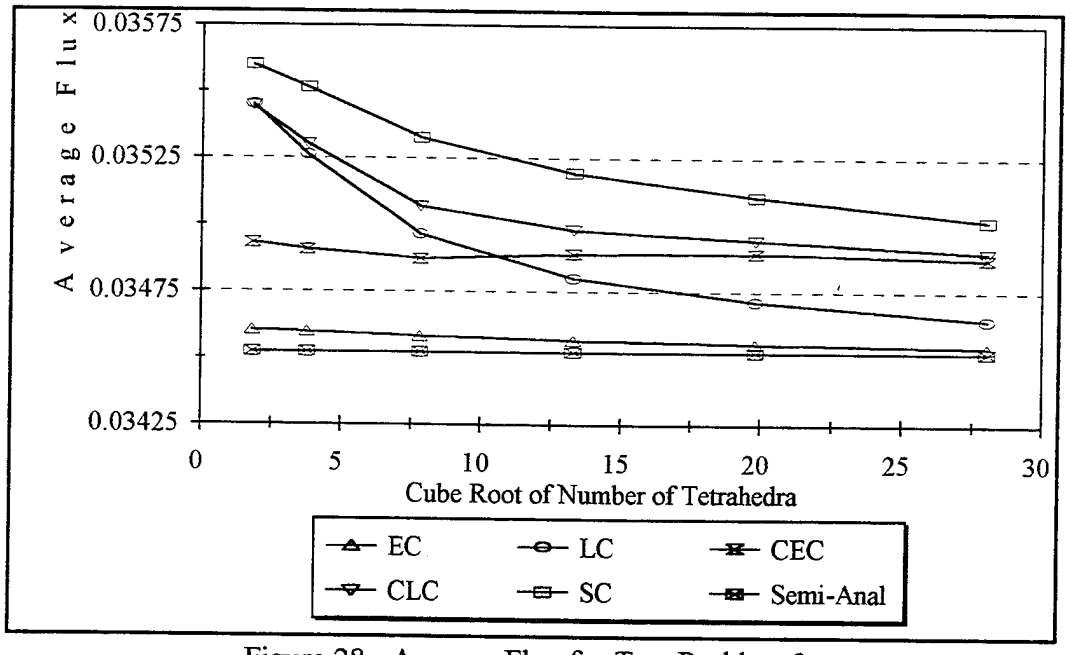

Figure 28. Average Flux for Test Problem 2.

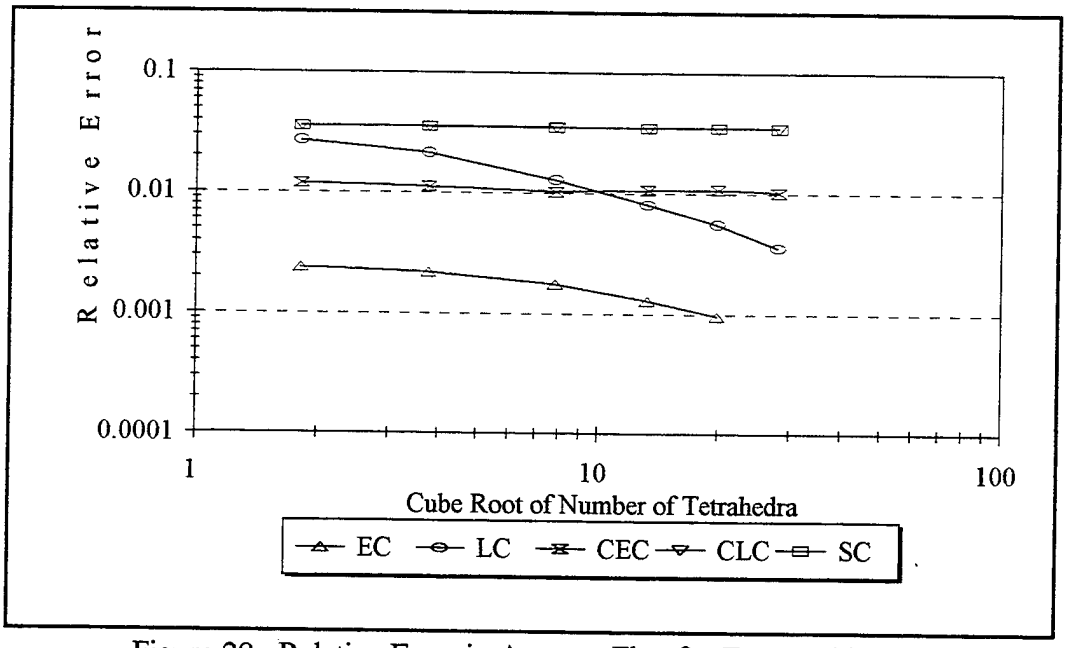

Figure 29. Relative Error in Average Flux for Test Problem 2.

The most stressing calculation is the exiting face (penetrating) current. The penetrating current and relative error of the penetrating current compared to the most converged EC solution are shown in Figure 30 and Figure 31. The magnitude of the exiting flux computed using the EC method is essentially constant for all mesh refinements. The LC solution oscillates between positive and negative values as the mesh is refined, and only approaches the EC result on the finest mesh. Examining the relative error plot, we see that the LC solution converges toward the correct result, but even the finest mesh LC result is not as accurate as the six tetrahedron EC solution. The exiting current predicted by the constant face methods differed by many orders of magnitude on the coarser meshes and is still off by two orders of magnitude on the finest mesh.

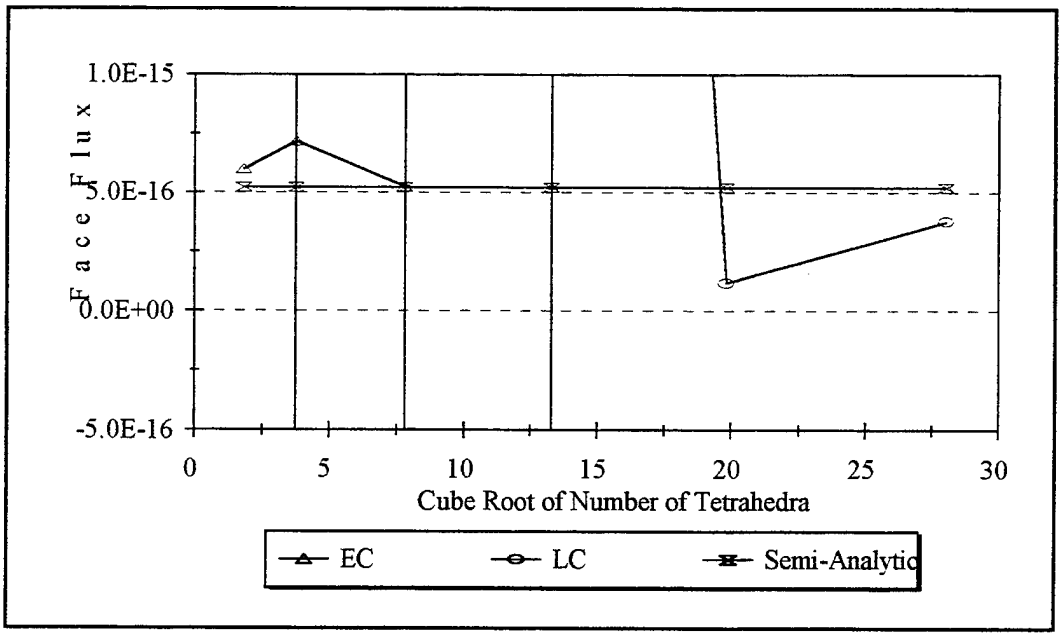

Figure 30. Exiting Face Current for Test Problem 2.

On the coarsest mesh, six tetrahedra, the relative errors in the EC solutions for the problem average flux and exiting current are less than LC solution on the finest mesh (22028 tetrahedron). So even though the LC solution costs 1/8 as much per cell as the EC solution, the LC method requires 3375 times as many cells to approach the same accuracy. For this type of problem, the EC method is clearly the most computationally efficient.

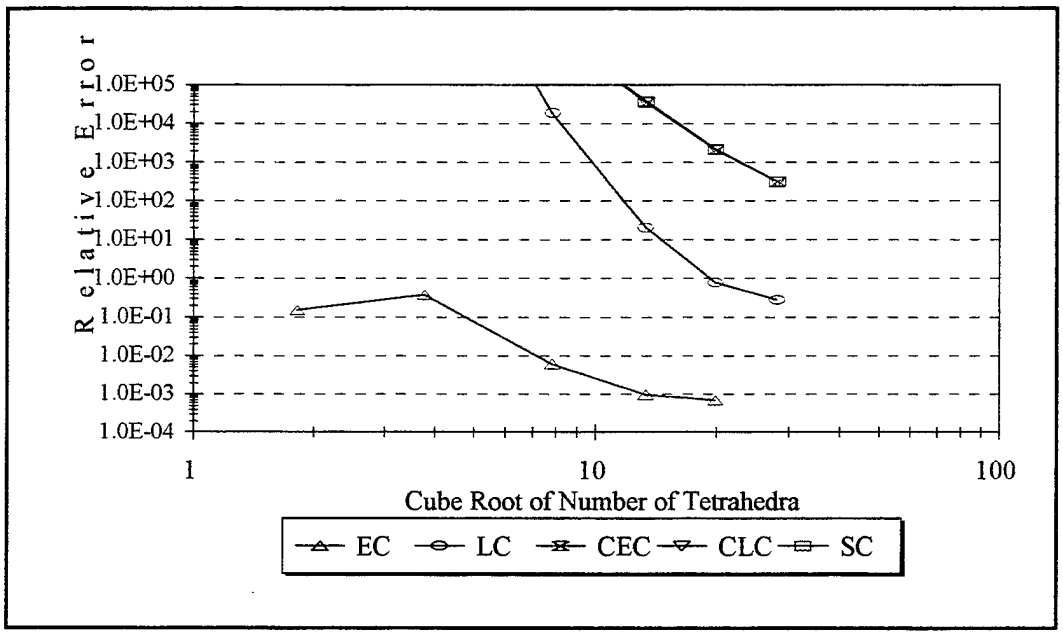

Figure 31. Relative Error in Exiting Face Current for Test Problem 2.

On the coarse meshes, LC gives unphysical negative exiting currents. The fact that the CLC method remains positive indicates that it is the face distribution that causes the LC method to generate negative exiting currents. This demonstrates the idea that CLC is essentially a negative face flux fix up for LC. Nevertheless, the LC solutions are more

accurate than the CLC solutions, which raises the issue of positivity versus accuracy. The LC quadrature produces local negative values, but the integral quantities, problem average flux and exiting face current, are more accurate than the strictly positive (SC, CEC) constant face method solutions. Further, exiting face flux and current values produced by the constant face methods that differ by several orders of magnitude from the correct value are no more useful than the negative values produced by the LC method on those meshes. In fact, the negative LC values may be less problematic because they are obviously nonsense, whereas the constant face results could be interpreted as being significant. Strict positivity in and of itself is not necessarily a great advantage in the absence of other desirable features. Any advantages gained by the strictly positive behavior of the constant face methods in this problem are offset by the increased accuracy of the first-moment conserving LC quadrature, while SC and LC are overshadowed by EC, which is both accurate and strictly positive.

Table 3 compares the benchmark and the finest mesh (22028 cell) TetSN EC solution values for several integral quantities. The agreement between the solutions is very good; all listed quantities agree to at least three digits. Although the differences are small, they are significant since the TetSN solutions were converged to six digits. The errors in the TetSN solution are most likely caused by numerical diffusion. Carrying first moments on faces, which the first moment methods do, reduces numerical diffusion, but does not eliminate it. The entering current computed by TetSN is slightly higher than the

benchmark, meaning fewer particles were back scattered at the entering face. A small fraction of the particles are numerically diffusing into the problem interior. Because fewer particles are scattered back at the face, the TetSN entering face flux is lower, and problem average flux and exiting face flux and current are higher, than the benchmark solution.

Table 3. Comparison of One-Dimensional Semi-Analytic and TETSN Results for Integral Quantities of Test Problem 2.

Quantity	1D Semi-Analytic Result	TETSN Result	Relative Error
ϕ_A	0.034523	0.034545	6.37×10^{-4}
ϕ (0)	2.8513	2.84097	3.62×10^{-3}
ϕ (1)	5.8095×10^{-16}	5.8133×10^{-16}	6.54×10^{-4}
J^{net} (0)	0.96662	0.96725	6.52×10^{-4}
J^{net} (1)	5.2159×10^{-16}	5.2181×10^{-16}	4.22×10^{-4}

Note that since the flux is decreased by 16 orders of magnitude in traversing the shield, only one in 10^{16} particles started in a Monte Carlo solution would reach the exiting side and contribute to a tally. However, because of the symmetries involved, certain variance reduction techniques could be employed that would generate an efficient and accurate Monte Carlo solution for the test problem 2 exiting flux with a modest number of histories. Difficulties arise when spatial information about fluxes and currents are required, especially if there are no symmetries to exploit. If spatial distributions are needed then separate estimators (tallies) are needed at each point information is desired.

Each new estimator added to a computation increases the computational effort. Further, the variance reduction techniques appropriate for one estimator are not necessarily helpful, and in fact may be detrimental, in reducing the variance of other estimators in the problem. For more complicated structures in which spatial distributions are required, the computation costs may rapidly become prohibitive.

5.3. Test Problem 3 - Cylinder-in-a-Cube, Isotropic Cylinder Source

Test problem 3 consists of a 2x2x2 cm cube with total cross section of 4 cm^{-1} and scattering cross section of 2 cm^{-1}. Embedded in the center of the cube is a 0.5 cm radius, 1 cm long cylinder. The cylinder material has a total cross section of 1 cm^{-1} and a scattering cross section of 0.5 cm^{-1}. In addition, the cylinder material contains an embedded isotropic source of strength 1 cm^{-3} sec^{-1}. A cutaway view of the 2511 tetrahedron mesh for test problem 3 is shown in Figure 32. The TetSN solutions generated in this section use an S_{16} level symmetric quadrature and a convergence tolerance of 10^{-6}.

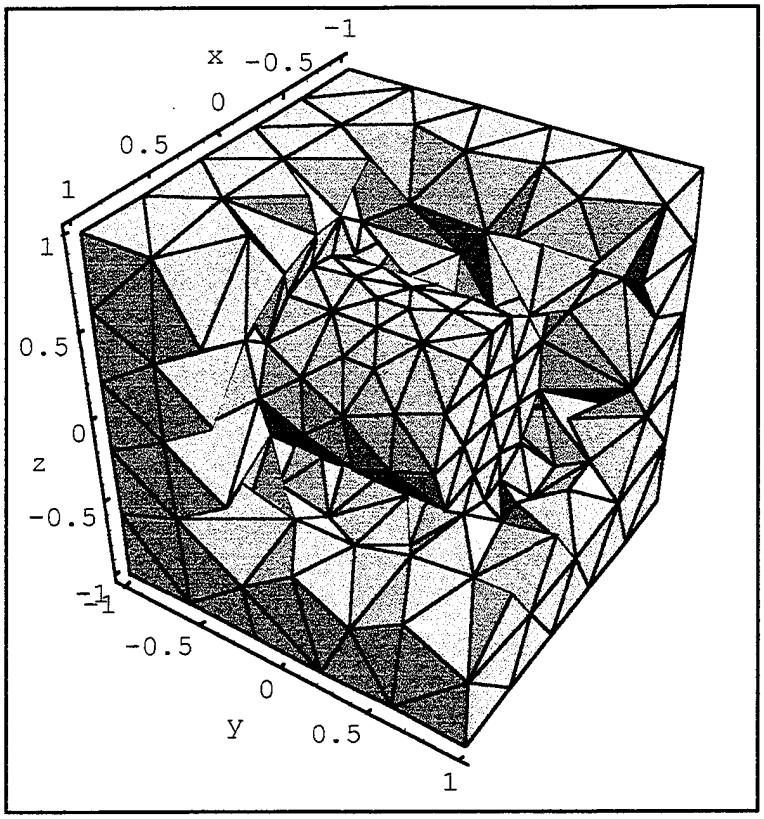

Figure 32. Cut Away View of the Test Problem 3 2511 Cell Mesh.

Interpreting the convergence for this problem is not as simple as in the previous cases. For any problem involving curved surfaces, such as the cylinder in this problem, refining the mesh also changes the region volumes. When IDEAS meshes a curved surface, it picks points on the surface of, and interior to, the volume, and creates cells by connecting those points. The result of this is that volumes of regions with concave boundaries are overestimated, and volumes of regions with convex boundaries are underestimated, but all volumes approach the correct volume as the mesh is refined. It is possible to compensate for this by repositioning the surface , but without prior knowledge of exactly how IDEAS is going to generate cells, there will always be some error. Table 4 lists the cylinder volume and error relative to the exact volume for all mesh refinements.

Table 4. Volume of Mesh Representation of Cylinder (Source Region) for Test Problem 3 and Error Relative to the Exact Volume (0.78539 cm^3).

# Cylinder Cells	# Cube Cells	Total Cells	Cylinder Volume	% Error
55	284	339	0.69207	11.88
174	731	905	0.73240	6.74
366	1432	1798	0.75689	3.63
593	1918	2511	0.76412	2.71
1272	5497	6769	0.77238	1.66
1844	9597	11441	0.77611	1.18

The problem, cube material, and cylinder material average fluxes are shown in Figure 33 through Figure 35, along with the MCNP results. The MCNP uncertainty for each of the average fluxes is 0.02%. Several trends are apparent. First, in each plot, the difference between the first order and constant face solutions is obvious. The constant face methods, as they have in every test problem, converge more slowly to the solution. However, on the finest mesh the constant face relative error with respect to the MCNP result is 3%, only slightly larger than the 1.3% first moment method error.

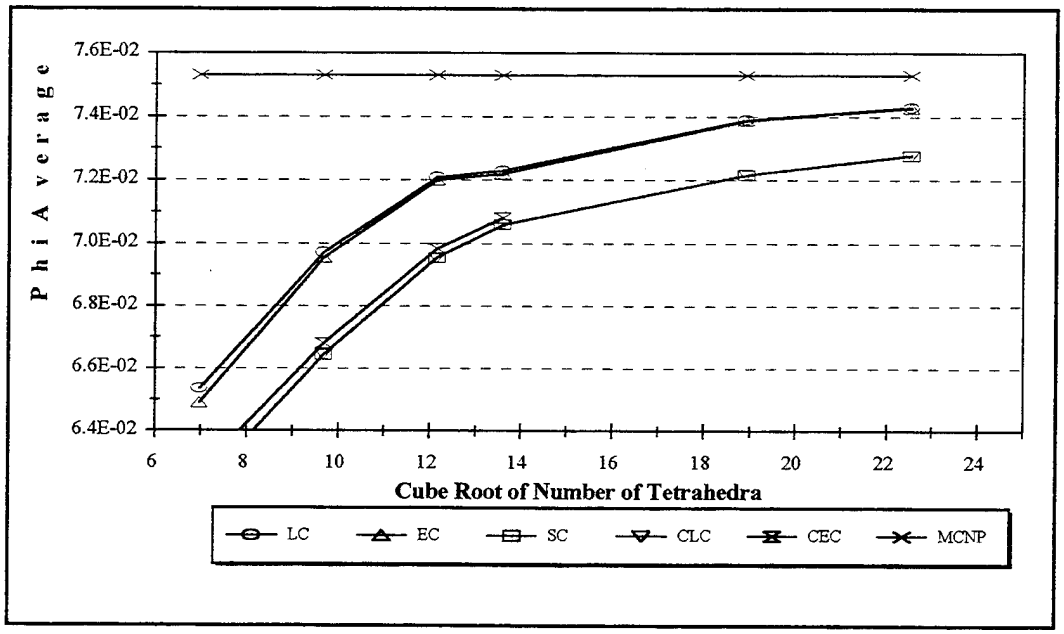

Figure 33. Problem Average Flux for Test Problem 3.

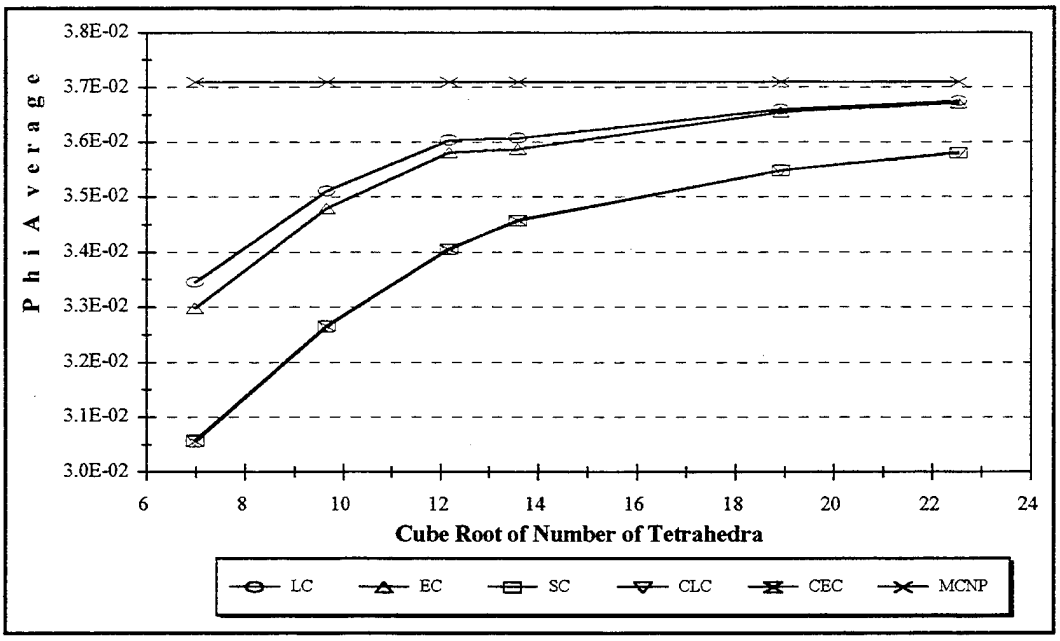

Figure 34. Cube Material Average Flux for Test Problem 3.

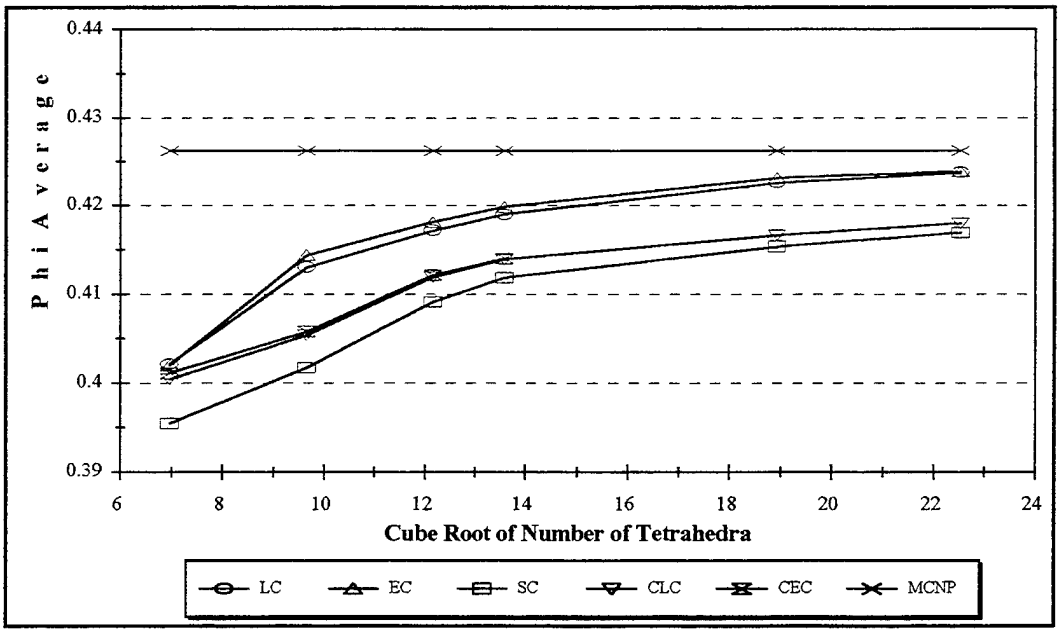

Figure 35. Cylinder Material Average Flux for Test Problem 3.

The difference in the constant face and first moment method solutions is larger in Figure 36 and Figure 37, plots of the exiting currents on faces parallel and perpendicular to the axis of symmetry of the source cylinder. The MCNP uncertainty for each exiting current is 0.2%. In this case, the EC method does a slightly better job computing the exiting current on coarser meshes, but both LC and EC reach the same solution on the finest mesh. Again, the constant face solutions converge much more slowly. The finest

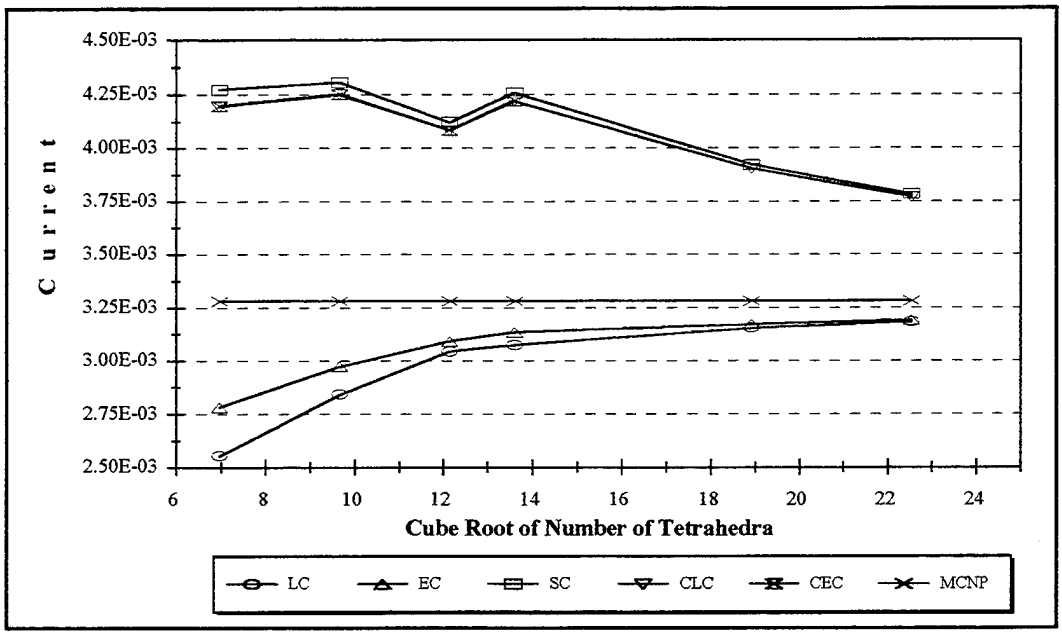

Figure 36. Exiting Current at z = -1 cm for Test Problem 3.

mesh constant face solution for the exiting face flux has about the same error relative to the MCNP solution, 15%, as the coarsest mesh EC result. To make any comparison of the TetSN and MCNP solutions at any one refinement is difficult because of the source volume error in the TetSN mesh. However, as the mesh is refined, all solutions approach the MCNP solution, indicating that the code is operating correctly.

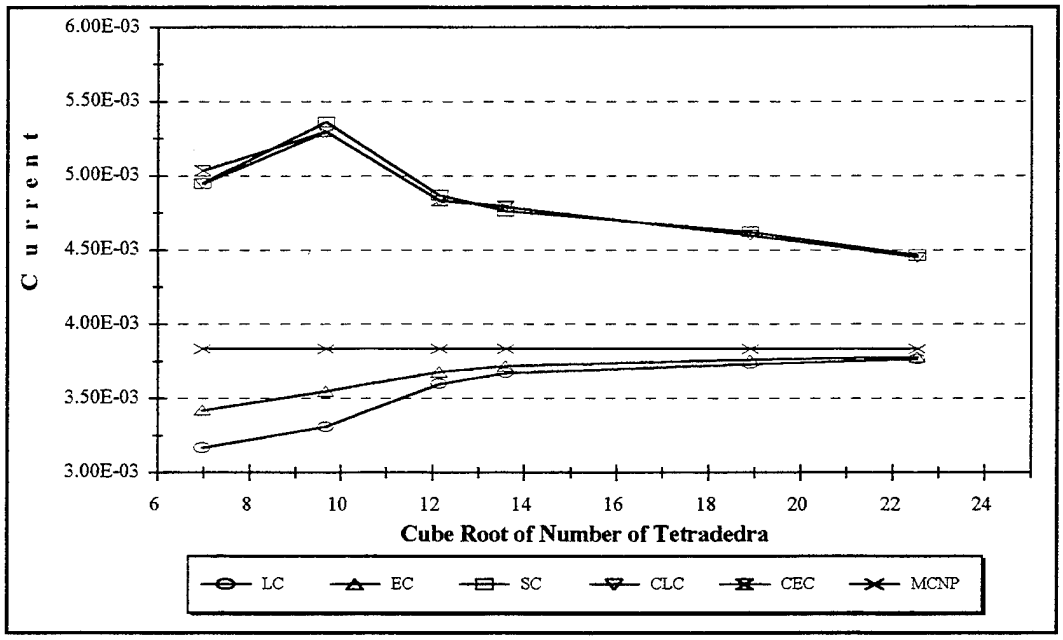

Figure 37. Exiting Current at y = -1 cm for Test Problem 3.

The cylindrical source drives all the fluxes and currents in the problem. Part of the error in the TetSN solution at each mesh point is due to the error in the source volume. In essence, because the source volume changes, each mesh refinement generates a slightly different problem. This masks the true behavior of the quadratures. Corrected values for the fluxes and currents can be computed by multiplying the TetSN results by the ratio of the correct cylinder volume to the meshed cylinder volume. Plots of both the corrected and uncorrected values for the problem average flux and exiting current are shown in Figure 38 and Figure 39. The relative errors in the corrected solutions are much lower than in the uncorrected case. The worst case error in the first order solution for the average flux drops from about 13% to 3%, and the worst case error in the exiting current drops from 23% to about 12%.

It is not clear that this volume correction approach works in general. If we were to plot corrected values of the exiting flux computed with the constant face methods, shown in Figure 37 and Figure 36, the relative errors would actually increase. A more consistent approach may be to position the problem surfaces so that the meshed volumes better approximate the actual region volumes. However , as already pointed out, this requires some advance knowledge of how IDEAS will generate the cells, which may not always be obvious. Further, this approach may prove to be very tedious because the surfaces have to be repositioned before generating the cells for each refinement. A better solution to volume correcting the results may be to devise a mesh generator that automatically conserves volumes.

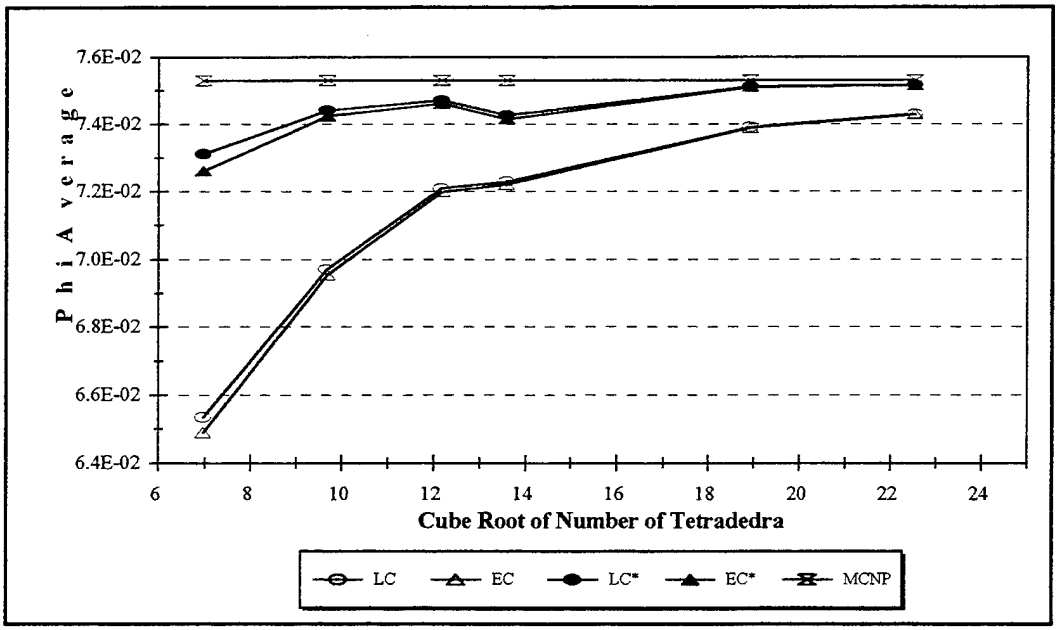

Figure 38. Problem Average Flux for Test Problem 3. * Indicates Results are Corrected for the Source Volume.

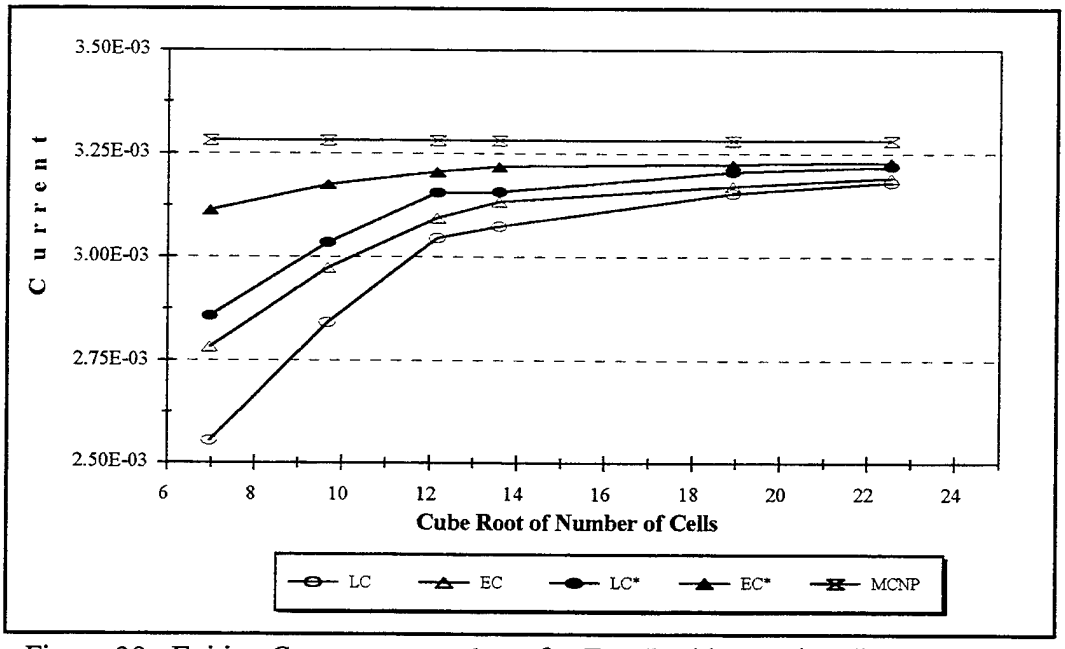

Figure 39. Exiting Current at z = -1 cm for Test Problem 3. * Indicates Results are Corrected for Source Volume.

5.4. Test Problem 4 - Cylinder-in-a-Cube, Isotropic Incident Current

Test problem 4 consists of an infinite slab cell with a periodic array of embedded, cylindrical absorbers. The calculation is performed on a unit cell of the array, giving the same cylinder in a cube structure used in test problem 3 but with different boundary conditions. The isotropic source in the cylinder is replaced by an isotropic current incident at z=-1 cm. The cube material has a scattering cross section of 7 cm^{-1} and a total cross section of 10 cm^{-1}. The cylinder is a strongly absorbing material with scattering cross section of 2 cm^{-1} and total cross section of 20 cm^{-1}. The symmetry boundaries exist at $x = \pm 1$ cm and $y = \pm 1$ cm , vacuum boundaries exist at $z = \pm 1$ cm.

This problem is a modified version of the deep shield problem evaluated in test problem 2. Since we already know that the constant face methods perform poorly on this type of problem, we concentrate on the first moment methods. The problem average flux computed using the LC and EC spatial quadratures is shown in Figure 40. The average flux in the cube and cylinder materials is shown in Figure 41 and Figure 42. In all cases, the EC result is essentially converged on the coarsest mesh. The LC quadrature is able to achieve results accurate to within 6% of the converged result for the problem and cube material average fluxes on fairly coarse meshes. This is again due to the average flux being dominated by the boundary layer near the entering face. The flux away from the boundary layer contributes only a small percent to the average flux. Since the EC method better approximates the actual distribution deep within the shield even on coarse meshes, it is able to correctly account for the deep shield contribution for all mesh refinements.

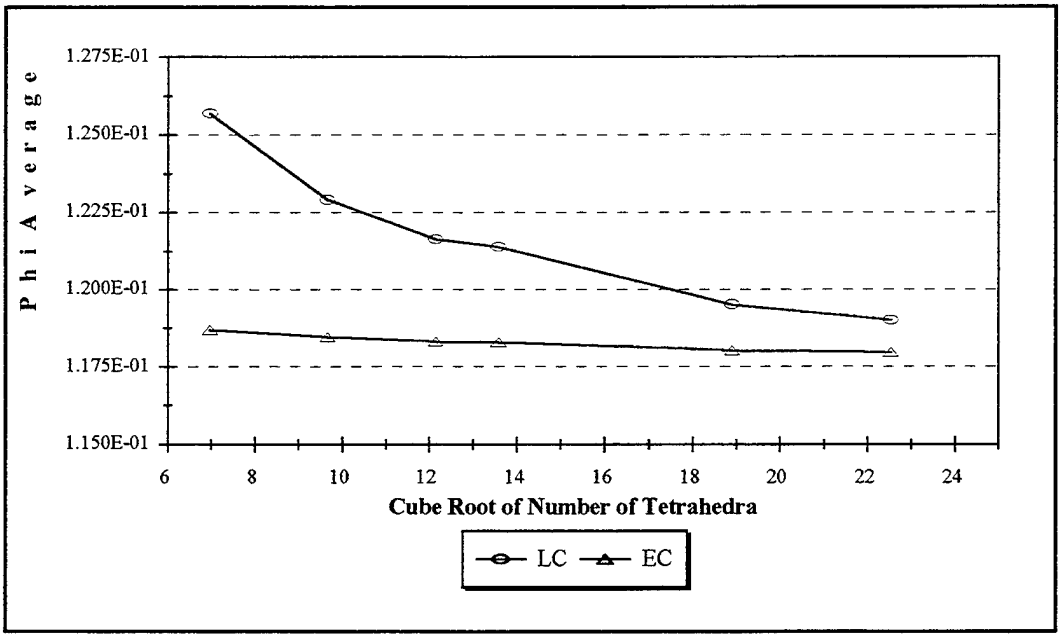

Figure 41. Problem Average Flux for Test Problem 4.

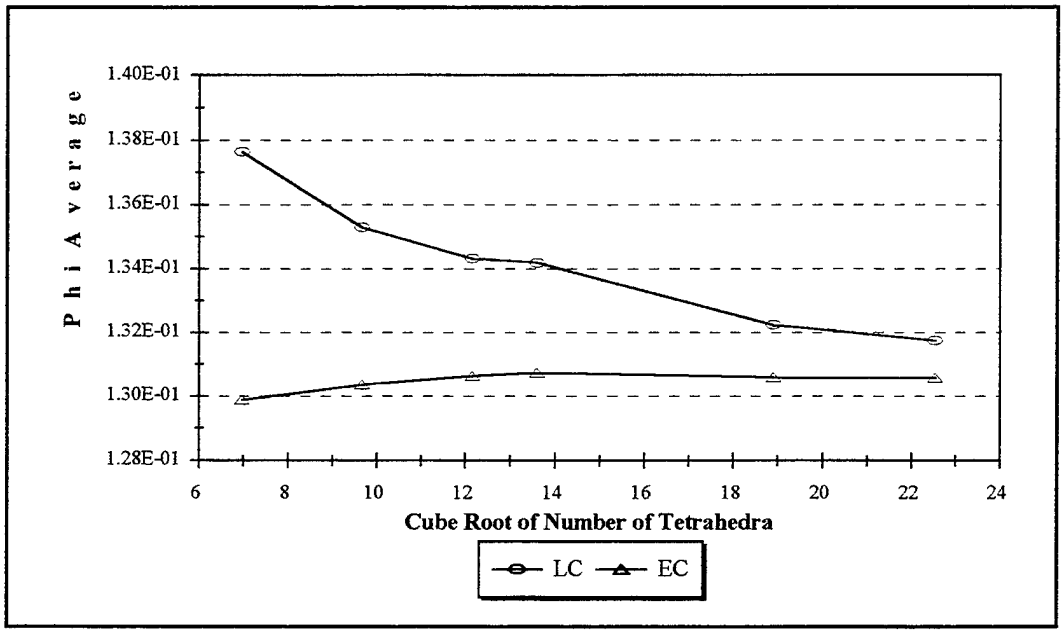

Figure 40. Cube Material Average Flux for Test Problem 4.

The LC method does not perform as well in computing the cylinder material average flux until the mesh is fairly refined. While the coarsest mesh (339 cells) LC results for the problem and cube average flux differ from the fully converged EC result by about 6%, the method required 11441 cells to reach the same level of accuracy for the cylinder average flux. This is again because the particles contributing to the cylinder flux have to traverse part of the cube material, where they are strongly attenuated. Upon reaching the cylinder, they are attenuated even more strongly. There is a boundary layer at the leading edge of the cylinder material, but it is not as large, nor as significant as the entering face boundary layer. For the LC method to compute the cylinder flux correctly, the mesh must be fine enough to allow the method to accurately model the flux fall off deep in the shield.

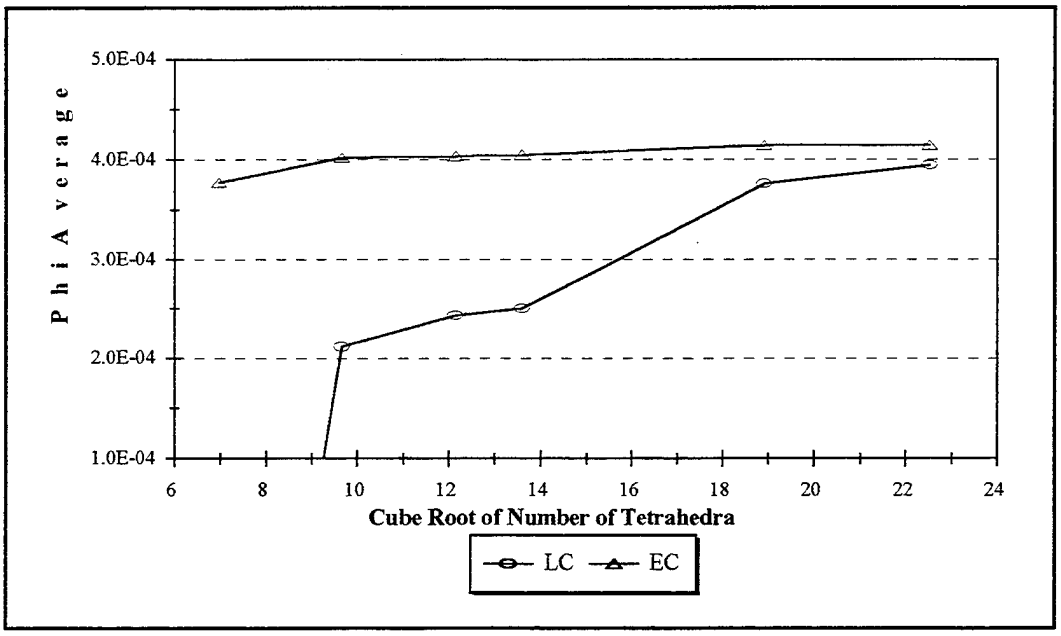

Figure 42. Cylinder Material Average Flux for Test Problem 4.

The most stressing calculation is the exiting face current. The average exiting face current computed by the quadratures at each refinement is plotted in Figure 43. Again, the EC method is essentially converged on the coarsest mesh, while the LC method overestimates the current by several orders of magnitude. The LC result still shows significant error, 12%, on the finest mesh. The main difference between this problem and test problem 2 is the presence of the absorber. In test problem 2, the exiting flux distribution was constant. In this case, the exiting flux has a nonuniform spatial distribution due to the presence of the absorber. Computing the exiting flux distribution is especially difficult in this case because the flux is already highly attenuated by the shield

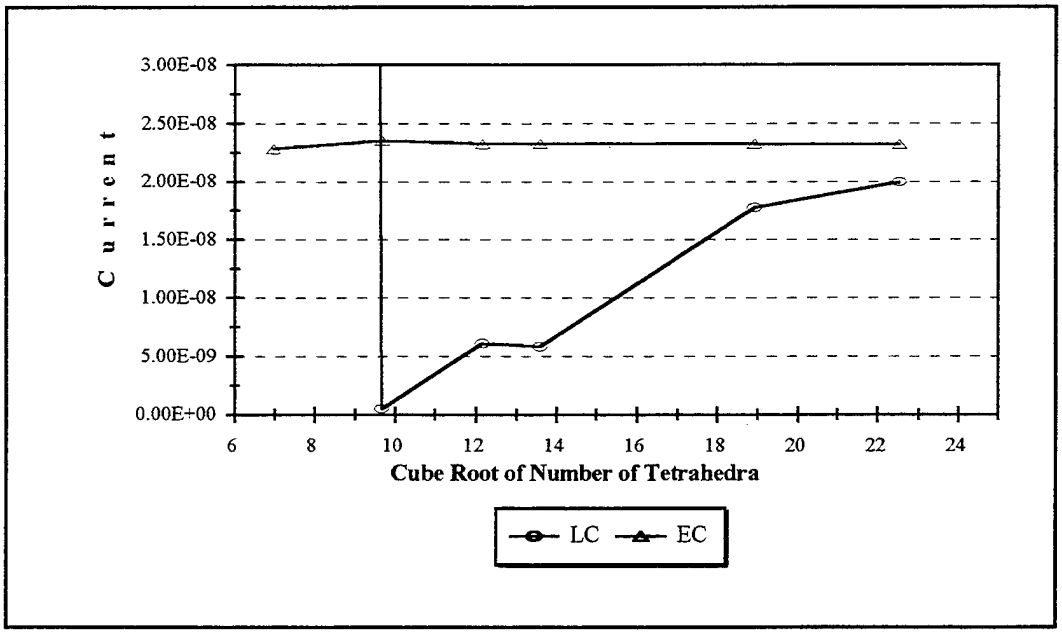

Figure 43. Exiting Face Current for Test Problem 4.

(cube) material. Without the cylindrical absorber, the flux would decrease many orders of magnitude in traversing the shield. The addition of the cylindrical absorber makes a small, but significant, change in the exiting flux distribution.

The exiting flux distributions computed using the EC and LC methods for each refinement are plotted in Figure 44 and Figure 45. The EC quadrature senses the presence of the absorber, and has essentially the correct distribution on the coarsest mesh. On the finest mesh, the EC method smoothly models the distribution, and the slope of the distribution on the boundary (since the problem contains an infinite array of embedded cylinders, the derivative of the face flux at $x=\pm 1$ and $y=\pm 1$ must be zero). The LC method is only able to resolve the flux dip on the finest mesh, and is never able to get the slope on the face edges.

Figure 44 and Figure 45 show the flux values plotted using the mesh nodes defining each face, and the flux distribution coefficients for the assumed form of the distribution. For the linear distribution, this shows all the available detail. For the EC solution, the solution on each face is exponential, and there is detail internal to each face as shown in Figure 46. This shows that EC does an even better job of modeling the face flux on the coarse mesh than Figure 44 would indicate.

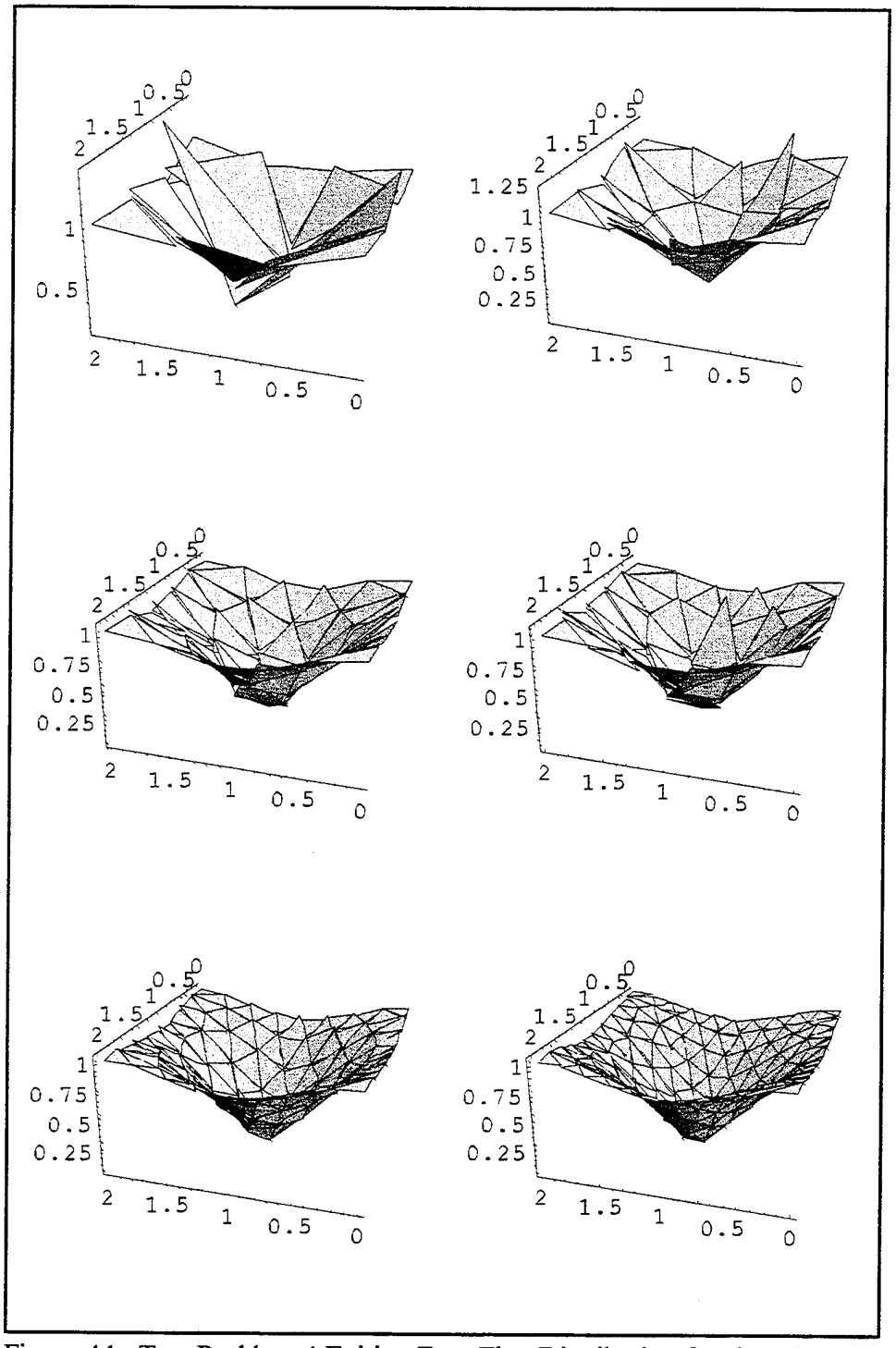

Figure 44. Test Problem 4 Exiting Face Flux Distribution for the EC Method (vertical-axis is flux * $2x10^7$)

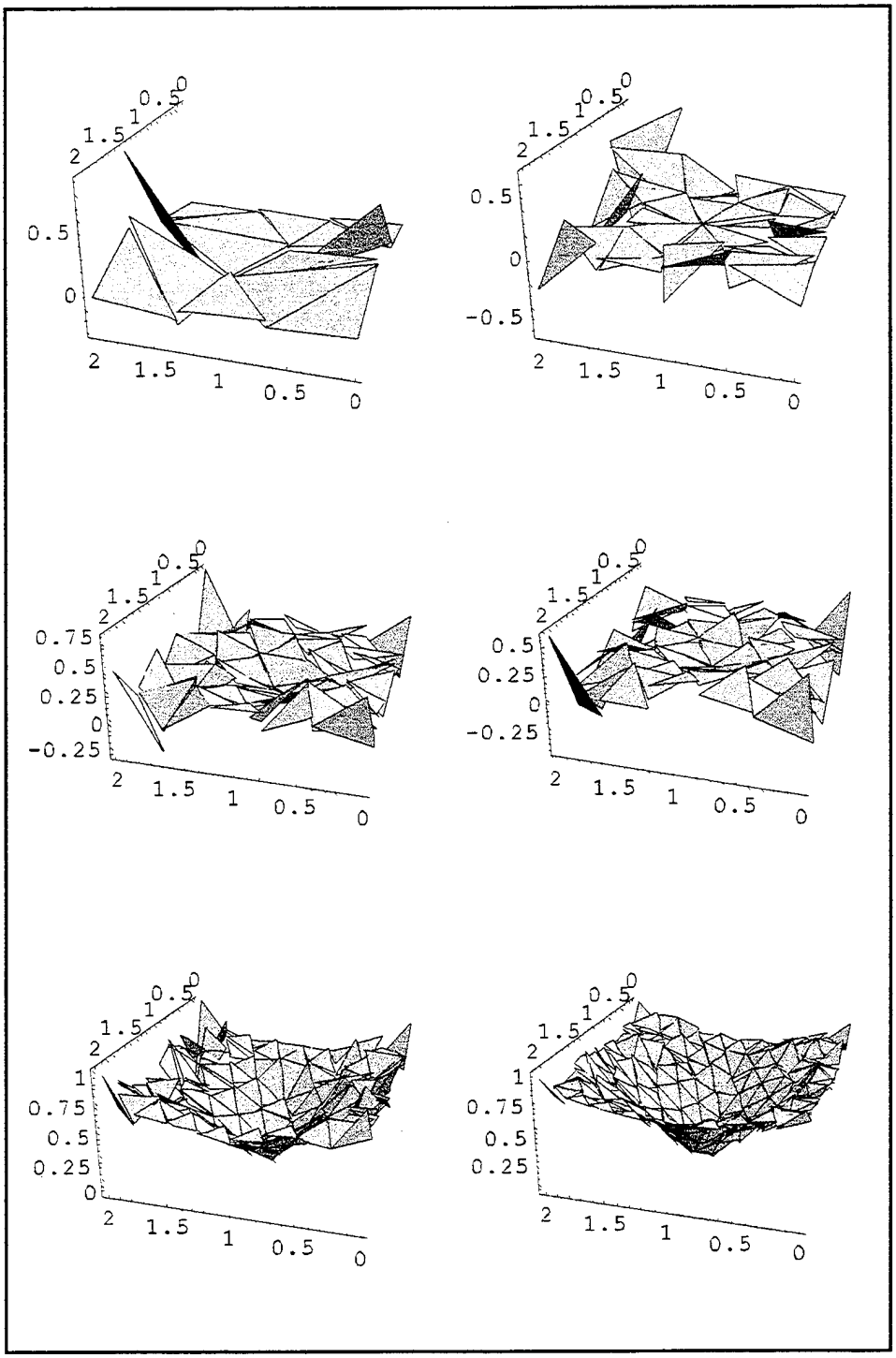

Figure 45. Test Problem 4 Exiting Face Flux Distribution for the LC Method (vertical-axis is flux * $2x10^7$)

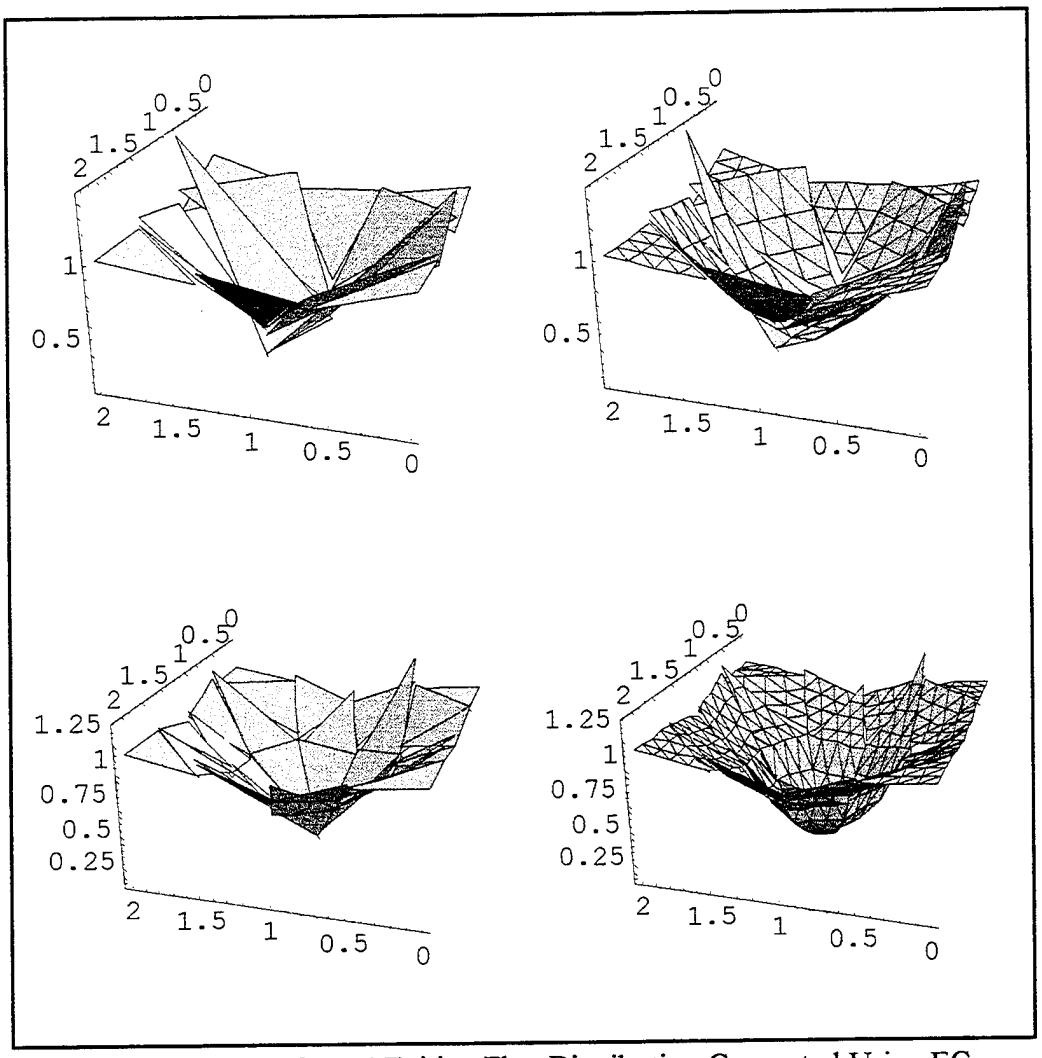

Figure 46. Test Case 4 Exiting Flux Distribution Computed Using EC on Two Coarsest Meshes Using Nodes Values (Left Side) and Using Internal Data Points (Right Side). (vertical-axis is flux*2x10^7)

A quantitative analysis of the convergence of the exiting face distribution was done using the error norm analysis described in Section 5.1.1. The L_∞ and L_1 norms of the errors in the exiting face distribution are shown in Figure 47 and Figure 47. While the maximum error in the LC solution decreases rapidly with decreasing cell size, the EC error

for each refinement is still significantly less. The average error in the coarsest mesh EC solution is less than the finest mesh LC solution. In this case the LC method requires more than 37 times as much computational effort to achieve the same accuracy in the L_1 norm as the EC method. This demonstrates the advantages of the EC method in deep shielding problems for pointwise as well as integral measures.

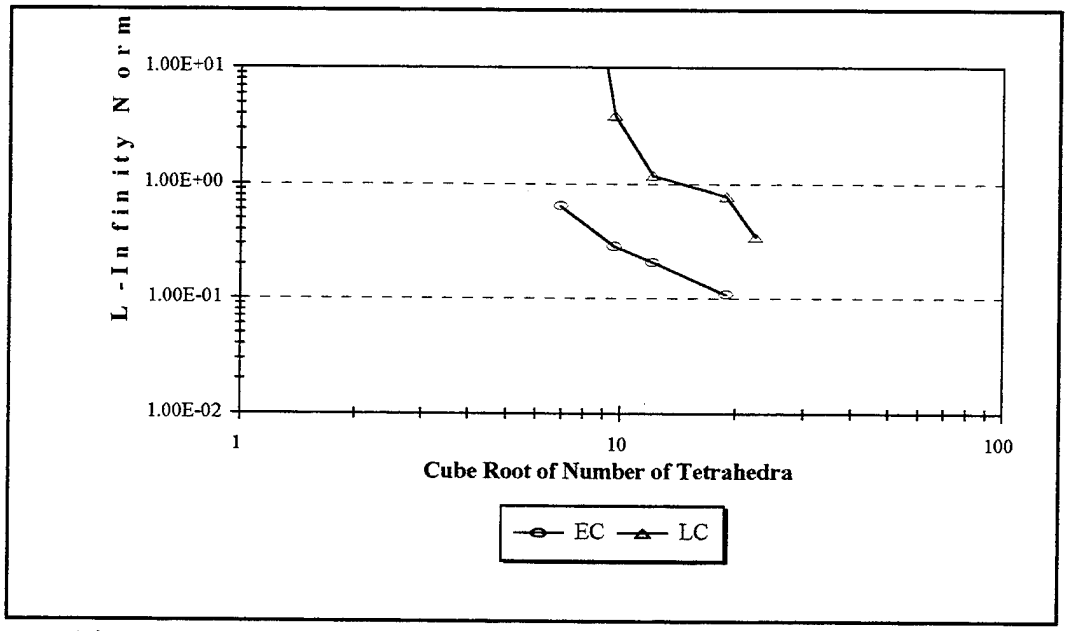

Figure 47. Maximum Pointwise Relative Error of the Exiting Face Flux Distribution for Test Problem 4.

Again, as in test problem 2, a Monte Carlo method would have a very difficult time obtaining results for the average exiting current because of the probability of the particles traversing the shield. Computing the exiting current distribution is even more stressing because the Monte Carlo method would then have to bin the surface into many sub-

surfaces. Then each sub-surface would require as many particles to compute its average current as the entire face required to compute the average.

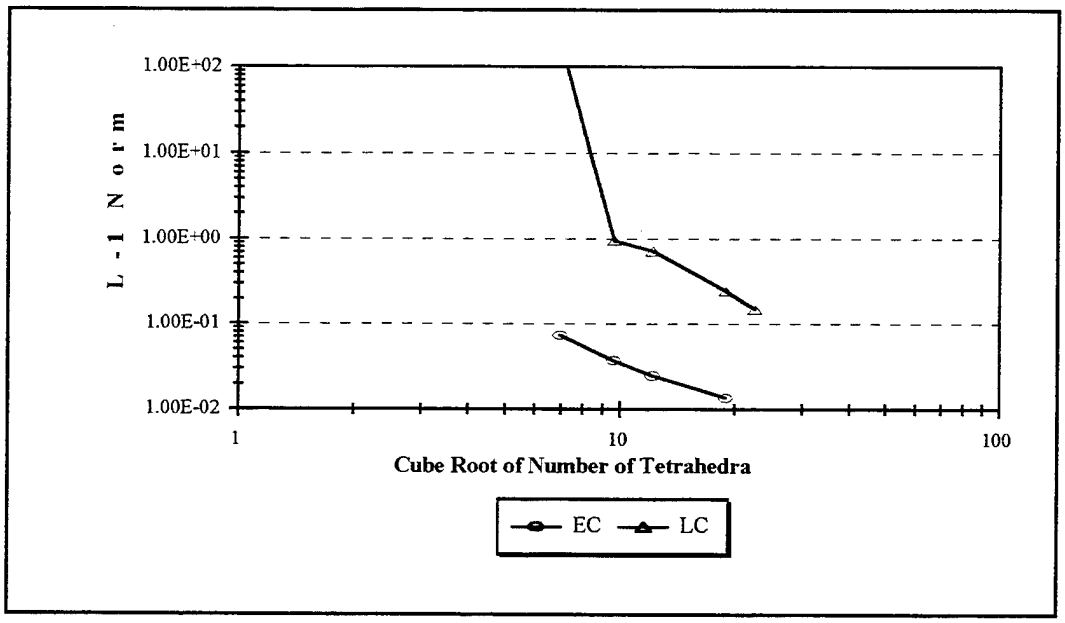

Figure 48. Average Pointwise Relative Error of the Exiting Face Flux Distribution for Test Problem 4.

VI. Summary/Recommendations

Characteristic spatial quadratures for performing discrete ordinates calculations using arbitrary tetrahedral meshes were derived, implemented, and tested. Tetrahedral meshes offer several advantages over traditional boxoid meshes. First, inclined or curved surfaces can be accurately modeled using fewer cells. More importantly, meshes can be refined in localized areas of the problem without drastically affecting the cell sizes in the rest of the problem, as is required for rectangular parallelpiped cells.

The algorithm used a split cell approach that reduced the BTE to a single dimension, simplifying the characteristic solution and the resulting moments equations. Implementation of the algorithm required that the methods be derived for the special case tetrahedron having a single inflow and a single outflow face. This included developing routines that could numerically evaluate exponential moments functions with four arguments. Methodology for defining and passing moments of the face flux distribution were also developed. For the EC method, two- and three-dimensional root solvers were developed using both single and multiple argument formulations to increase efficiency while assuring stability. Equations to provide an initial guess which guaranteed convergence of Newton' s method in fewer than four (generally fewer than three) iterations for the face flux and source distribution systems were also developed.

Monte Carlo methods have become the preferred method for doing general-geometry three-dimensional neutral particle transport due both to the inherent benefits of

the Monte Carlo methods and the lack of an efficient general-geometry 3D discrete ordinates code. Aside from some special purpose codes designed for specific applications, 3D discrete ordinates codes are limited to low order spatial quadratures and rectangular parallelpiped cells. Although there are many problems for which Monte Carlo methods are an appropriate choice, there are many situations, particularly when the media are optically thick or when spatial distributions are required, when discrete ordinates may be more computationally efficient. For what ever the reason, the development of improved spatial quadratures has not been translated into the development of improved operational codes, particularly in three dimensions. The TetSN code is intended to be a first step toward the development of a general-geometry, three-dimensional code package that will provide a needed computational tool to the transport community.

6.1 Summary of Results

For test problems 1, 2, and 3, the characteristic quadrature solutions converged to the independent benchmark result, as both the spatial and angular variables were refined. Both the first moment methods, LC and EC, exhibited third order convergence, consistent with the convergence rates observed for the methods in one and two dimensional geometries. The constant face methods were all approximately first order convergent, again consistent with the SC method's performance in other geometries. The relatively slow convergence of the constant face methods on fine meshes, and the resulting low accuracy and severe numerical diffusion on coarse meshes, severely limits their use in practice. Based on the cases examined, there appears to be no benefit whatsoever of using

a constant face flux approximation with a higher order source distribution approximation. The SC quadrature performed as well as the CLC and CEC methods on each of the test problems.

All of the quadratures proved to be relatively insensitive to variations in the mesh structure. The results show that relative to the precision of the computed results, the fluctuations in solution caused by variations in the mesh structure were essentially insignificant. This is a very important quality. The meshes generated in Section 5.1.2, contained some very poorly shaped tetrahedra. In some situations, it may be necessary to model an oddly shaped region using tetrahedra with poor aspect ratios. For these problems it is essential that the computation method be insensitive to variations in the shape of the cells.

In terms of computation time per phase space cell, the EC method was the most expensive, requiring 13 times the effort of SC and eight times the effort of LC. However, the EC method excelled on problems with optically thick cells. For test problem 2, the EC method required 1/675 of the effort of the LC method to achieve the same accuracy. In test problem 4, where the cells were not as thick, the EC method required 1/37 of the computational effort of LC to compute the integral quantities, while the LC method never reached the coarse mesh EC accuracy in computing the exiting flux distribution.

On problems where the cell optical thickness was in the 1mfp range, the LC method was the best performer. In test problems 1 and 3, the EC and LC methods gave about the same results at each level of mesh refinement, but LC required only 1/8 of the

EC effort to achieve the same accuracy. However, the fact that the EC method provides accurate results on both thin and thick cells is a very desirable trait. Since in general multi-group problems, the optical thickness of a material can change several orders of magnitude based on the particular energy of the particles, it is important to have a method that is stable regardless of the cell size. In this respect the EC method was again the best performer.

The advantages of discrete ordinates over Monte Carlo solutions for optically thick problems were also considered. For Problems 2 and 4, where the flux dropped many orders of magnitude in traversing the shield, the Monte Carlo method would have required a prohibitive number of particles to compute the average results without considerable variance reduction assistance. To compute distributions, the Monte Carlo method requires many times more particles than are required to compute the average quantities, further compounding the difficulty. The flux and current distributions are naturally computed as part of the discrete ordinates solution.

Finally, the advantages of using tetrahedral meshes were observed. Just as in the triangle case for two dimensions, tetrahedral cells are better able to model curved surfaces. This is evidenced in test problems 3 and 4 where the tetrahedra were able to model the volume and general shape of the central cylindrical region with a relatively small number of cells. This will prove very beneficial in problems with many curved surfaces.

6.2 Recommendations for Future Efforts

In order for the code to be useful as a production tool, several limitations of the current implementation must be addressed. Most important is execution speed. Running the test problems required anywhere from several seconds to over a day to complete. In a production environment, this is unacceptable. The current implementation was intended to be straightforward and understandable. There are doubtless may opportunities for optimization including coding practices, compiler optimizations, and even operating system optimizations. The area in which modest investment of effort would pay the most dividends is in increasing the efficiency of the exponential moments functions routines. Currently the EC and LC methods spend about 60% and 30% respectively of their execution time in the moments function routines. All moment function evaluations eventually reduce to computing a number of single-argument functions. The single argument routines were written to be able to accurately compute functions of any order. However, as can be seen from the equations in Chapter 3, the overwhelming majority of the functions required are only of order 0 and 1. Currently, the routines for numerically evaluating the moments functions are able to treat all orders. By writing routines to deal only with zeroth and first order functions, it may be possible to gain considerable computational efficiency.

For the EC method, it may be possible to gain some computation speed by defining a new set of functions. As shown in Chapter 3, all the equations for the EC method first moments are in terms of the ratio of a first order to a zeroth order function of the same

arguments. As shown in Appendix B, the domain of these functions is bounded between zero and one. Rather than trying to compute the moments functions in the numerator and denominator, it may be more efficient to treat the ratio itself as a function, and develop routines to numerically evaluate these new functions.

Increasing the efficiency of the root-solvers is also necessary. Currently, the time spent in the root-solvers is only a fraction of the total EC execution time, but addition of anisotropic scatter makes the rootsolver dominant. The code currently assumes only isotropic scatter. For many problems of interest this assumption is too limiting. Adding the capability to handle anisotropic scatter is straightforward, but adds considerable computational effort. Currently, the source distribution coefficients only have to be computed once per iteration, since the source is independent of angle. Once the assumption of isotropic scatter is dropped, the source becomes a function of angle, and the source distribution coefficients have to be computed for each angle. Referring back to Table 2, Section 5.1.3, for an S_{16} quadrature (288 angles), the computation of the source distribution coefficients for the EC method goes from being almost insignificant, to dominating the computation.

There are several ways that the root solvers might be optimized. First, the initial guess equations can be refined. Currently, the root-solvers require 3.5 iterations on average to converge. Having better initial guess equations would reduce the required number of iterations. It may even be possible to generate a set of equations that could generate reasonable approximations to the coefficients without having to rootsolve.

Another possibility is to examine alternate numerical techniques. We chose Newton's method because it is quadratically convergent, and because we had the capability to analytically evaluate the Jacobian. There are methods that are slightly less than quadratically convergent, that use a secant approximation to the function derivatives that may be more computationally efficient.

The code as written can accommodate multiple energy groups, but doing so adds considerable computational effort. For a multi-group problem, the solution must converge for each energy group. Therefore, the execution times listed in Table 2 have to be multiplied by the number of energy groups in the problem for down scatter only, and several times the number of energy groups when up scatter is allowed. This underscores the need to improve the efficiency of the moment function and root-solving routines.

As the efficiency of the moment function and root-solve routines increase, the splitting and assembling routines will become a larger fraction of the execution time. For the non-exponential methods, splitting and assembling already represent more than 50% of their execution time, hence improving the routines could considerably improve the performance of these methods.

Finally, although the IDEAS code was able to generate the tetrahedral meshes used in the test problems with a modest level of manual intervention, more complicated structures will be quite tedious to generate. Regardless of the shape of the object, the code can generate a tetrahedral mesh to fill it. However, when the geometry includes not only complex shapes, but several regions with different materials as well, it takes

considerable manual intervention to generate the mesh. None of these adjustments are obvious unless the operator is very familiar with the features of the code. Before TetSN will be useful as a production tool, a more user friendly approach to mesh generation must be devised.

Appendix A. Exponential Moment Functions

The exponential moment functions were introduced in section 3.3. This Appendix presents the methods used to numerically evaluate the functions.

A.1 One-Dimensional Exponential Moment Functions

Algorithms for numerically evaluating moments functions of a single argument were developed by Minor (1993). These routines are the basis of calculation for higher dimension moment functions.

A.2 Two-Dimensional Exponential Moment Functions

Expressed in terms of recursion on number of arguments, the form of the two-argument moment function is

$$M_n(x_1, x_2) = \frac{M_n(x_1) - M_n(x_2)}{(x_2 - x_1)}, \tag{118}$$

where $x_2 \neq x_1$. In general, any two argument moment function can be efficiently computed using this definition and the single argument routines developed by Minor. However, as the value of x_1 approaches that of x_2, the equation becomes numerically ill conditioned. When this occurs, Equation (118) can be rewritten as

$$M_n(w - \Delta w, w + \Delta w) = \frac{M_n(w - \Delta w) - M_n(w + \Delta w)}{2\Delta w}, \tag{119}$$

where $w = (x_1 + x_2)/2$ and $\Delta w = w - x_1$. In this form, as $x_1 \to x_2$, $w \to 0$, and the function can be approximated using a series expansion about $\Delta w = 0$.

In view of the orderless property, without loss of generality, any first partial derivative of an exponential moment function is obtained using

$$\frac{\partial}{\partial x_j} M_n(x_1, \ldots, x_m) = -M_n(x_1, \ldots, x_m, x_m) . \tag{120}$$

For functions of a single argument, the derivative is obtained from the integral definition, equation (36):

$$\frac{d}{dx} M_n(x) = M_{n+1}(x) - M_n(x) . \tag{121}$$

Higher order derivatives of a single argument function are found by recursively applying Equation (121) giving,

$$\frac{d^k}{dx^k} M_n(x) = \frac{d^{k-1}}{dx^{k-1}} M_{n+1}(x) - \frac{d^{k-1}}{dx^{k-1}} M_n(x) . \tag{122}$$

Using Equation (122) the expansion of Equation (119) to $O(\Delta w)^6$ is

$$\begin{aligned} M_n(w - \Delta w, w + \Delta w) \approx\ & M_n(w) - M_{n+1}(w) \\ & + \left[M_n(w) - 3M_{n+1}(w) + 3M_{n+2}(w) - M_{n+3}(w) \right] \frac{\Delta w^2}{6} \\ & + [M_n(w) - 5M_{n+1}(w) + 10M_{n+2}(w) \\ & - 10M_{n+3}(w) + 5M_{n+4}(w) - M_{n+5}(w)] \frac{\Delta w^4}{120} . \end{aligned} \tag{123}$$

The expansion was evaluated numerically to determine the required order of expansion for a given moment function order, n, and variable separation, Δw. The criterion used was that the computed value contain at most one digit of accuracy less than the computation tolerance used to compute the single-argument functions. Based on these results, the algorithm for computing two-dimensional moment function is as follows:

If ($\Delta w < 10^{-3}$)

Compute moment function using expansion with up to third order terms

Else If ($10^{-3} < \Delta w < 10^{-2}$)

Compute moment function using expansion with up to fifth order terms

Else

Compute moment function using recursion on order and single-argument moments functions routines

Endif

The expansions and algorithms hold without regard to the sign of the individual arguments.

If the variables are far enough apart in magnitude, the two-dimensional moment function can be computed using Equation (118), which requires that $M_n(x)$ and $M_n(y)$ both be calculated in the one-dimensional routines. If the variables are close to one another, i.e. Δw is small, the two-dimensional function is evaluated using the expansion. Numerically evaluating the expansion requires computing at most $M_n(w)$ to $M_{n+5}(w)$. This turns out to be a much more efficient calculation in most cases because the one-dimensional routines only need to compute a given order, then use recursion to get the remaining order functions, e.g. compute $M_n(x)$ then use recursion to get $M_{n+1}(w)$ to

$M_{n+5}(w)$. Computing a single order then recursing is generally faster than computing two separate functions, so the moment function calculation is actually faster when using the expansion. This effect is even more pronounced in higher dimension functions.

A.3 Three-Dimensional Exponential Moment Functions

Expressed in terms of recursion on number of arguments, the three-dimensional moment functions are

$$M_n(x_1,x_2,x_3) = \frac{M_n(x_1,x_2) - M_n(x_2,x_3)}{(x_3 - x_1)}, \tag{124}$$

where, without loss of generality, the variables x_1, x_2, and x_3 are arranged in ascending order. In general any three-dimensional moment function can be computed using this definition and routines capable of computing two-dimensional moment functions. However, numerical ill conditioning occurs when the value of x_3 approaches that of x_1. In those situations, an alternate form of Equation (124), given by

$$M_n(w-\Delta w, w+\Delta y, w+\Delta w) = \frac{M_n(w-\Delta w, w+\Delta y) - M_n(w+\Delta y, w+\Delta w)}{2\Delta w} \tag{125}$$

where $w = \frac{x_1 + x_3}{2}$, $\Delta w = x_3 - w = w - x_1$, and $\Delta y = x_2 - w$. Since Δy is always less than Δw, as Δw approaches zero, Equation (125) can be approximated by an expansion about $\Delta w = 0$ and $\Delta y = 0$. Expanding about $\Delta w = 0$ and $\Delta y = 0$ and only keeping terms to order five in Δw, Δy, or products of Δu and Δy gives,

$$
\begin{aligned}
M_n(w-\Delta w, w+\Delta y, w+\Delta w) \approx{}& \frac{1}{2}\left[M_n(w) - 2M_{n+1}(w) + M_{n+2}(w)\right] \\
&+ \left[-M_n(w) + 3M_{n+1}(w) - 3M_{n+2}(w) + M_{n+3}(w)\right]\frac{\Delta y}{6} \\
&+ \left[M_n(w) - 4M_{n+1}(w) + 6M_{n+2}(w) - 4M_{n+3}(w) + M_{n+4}(w)\right]\frac{\Delta y^2 + \Delta w^2}{24} \\
&+ [\, -M_n(w) + 5M_{n+1}(w) - 10M_{n+2}(w) + 10M_{n+3}(w) \\
&\qquad - 5M_{n+4}(w) + M_{n+5}(w)\,]\frac{\Delta y^3 + \Delta w^2 \Delta y}{120} \\
&+ [\, M_n(w) - 6M_{n+1}(w) + 15M_{n+2}(w) - 20M_{n+3}(w) \\
&\qquad + 15M_{n+4}(w) - 6M_{n+5}(w) + M_{n+6}(w)\,]\frac{\Delta y^2 \Delta w^2}{720} \\
&+ [\, -M_n(w) + 7M_{n+1}(w) - 21M_{n+2}(w) + 35M_{n+3}(w) - 35M_{n+4}(w) \\
&\qquad + 21M_{n+5}(w) - 7M_{n+6}(w) + M_{n+7}(w)\,]\frac{\Delta y^3 \Delta w^2}{5040}.
\end{aligned}
\tag{126}
$$

As for the two-dimensional case, the expansions were evaluated numerically to determine the required order of expansion for a given moment function of order n, and variable separation Δw. The criteria used were the same as for the two-dimensional case. The algorithm for determining which method to use in computing three-dimensional moments functions is:

Sort variables in ascending order

If ($\Delta w < 10^{-3}$)

Compute moment function using expansion with up to third order terms

Else If ($10^{-3} < \Delta w < 10^{-2}$)

Compute moment function using expansion with up to fifth order terms

Else

Compute moment function using recursion on order

Endif

For $\Delta w > 10^{-2}$, recursion on order gives accurate enough results. The resulting two-argument functions can then be sent to the two-dimensional routines for evaluation. However, if the separation of the arguments of both two-dimensional functions are large enough that they can be computed by recursion to first kind functions, then one of the first kind functions ends up being computed twice as shown in Figure 49. The algorithm for computing three-argument functions tracks the variable separation at each step and ensures the minimal amount of computation.

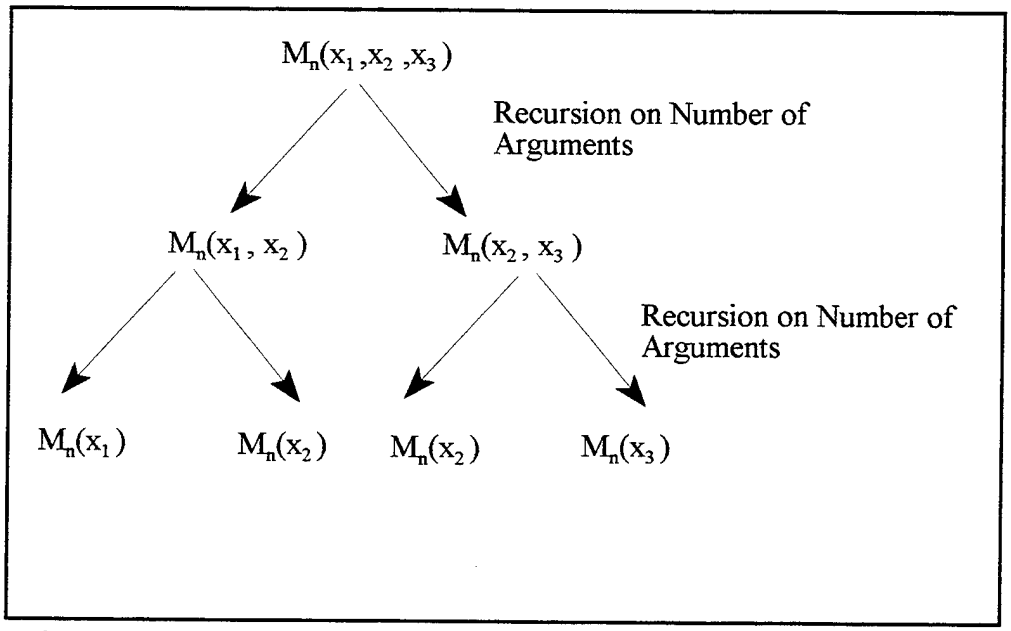

Figure 49 Details of the number of single moment functions that have to be evaluated to compute a three argument function by recursion on number of arguments.

With this in mind, the algorithm for computing three-argument moment functions is

Sort Variables in ascending order

If $(x_3 - x_1) < 10^{-2}$ *then*

Compute $M_n(x_1, x_2, x_3)$ *using expansion*

Else

If $(x_2 - x_1) < 10^{-2}$ *and* $(x_3 - x_2) < 10^{-2}$ *then*

Compute $Mn(x_1, x_2)$ *and* $Mn(x_2, x_3)$ *using expansions*

Compute $M_n(x_1, x_2, x_3)$ *using recursion*

Else If $(x_2 - x_1) < 10^{-2}$ *and* $(x_3 - x_2) > 10^{-2}$ *then*

Compute $Mn(x_1, x_2)$ *using expansions*

Compute $M_n(x_2)$, *and* $M_n(x_3)$

Compute $Mn(x_2, x_3)$ *using recursion*

Compute $M_n(x_1, x_2, x_3)$ *using recursion*

Else If $(x_2 - x_1) > 10^{-2}$ *and* $(x_3 - x_2) < 10^{-2}$ *then*

Compute $M_n(x_1)$, $M_n(x_2)$

Compute $Mn(x_1, x_2)$ *using recursion*

Compute $Mn(x_2, x_3)$ *using expansion*

Compute $M_n(x_1, x_2, x_3)$ *using recursion*

Else if $(x_2 - x_1) > 10^{-2}$ *and* $(x_3 - x_2) > 10^{-2}$ *then*

Compute $M_n(x_1)$, $M_n(x_2)$, *and* $M_n(x_3)$

$M_n(x_1, x_2) = (M_n(x_1) - M_n(x_2))/(x_2 - x_1)$

$Mn(x_2, x_3) = (M_n(x_2) - M_n(x_3))/(x_3 - x_2)$

Compute $M_n(x_1, x_2, x_3)$ *using recursion*

End If

End If

The expansions and algorithm are not dependent on the sign of the individual arguments.

If the arguments are such that the three-dimensional function can be computed using an expansion, then at most $M_n(w)$ to $M_{n+7}(w)$ will have to be computed. Any other situation requires that at least two, and at most three, separate single argument functions be evaluated. Since the calculation by recursion on order is very inexpensive, evaluation of the function using the expansion tends to be much quicker than evaluation by recursion or combination of recursion and expansion.

A.4 Four-Dimensional Exponential Moment Functions

Expressed in terms of recursion on number of arguments, the four-dimensional exponential moments functions are

$$M_n(x_1,x_2,x_3,x_4) = \frac{M_n(x_1,x_2,x_3) - M_n(x_2,x_3,x_4)}{(x_4 - x_1)} \tag{127}$$

where the variables are arranged in ascending order. In general any moments function having four arguments can be computed using this equation and routines that compute three-argument moment functions. Numerical ill-conditioning occurs as the value of x_1 approaches that of x_4. In this case Equation (127) can be written as,

$$M_n(w-\Delta w, w+\Delta v-\Delta u, w+\Delta v+\Delta u, w+\Delta w)$$

$$= \frac{M_n(w-\Delta w, w+\Delta v-\Delta u, w+\Delta v+\Delta u) - M_n(w+\Delta v-\Delta u, w+\Delta v+\Delta u, w+\Delta w)}{2\Delta w}, \tag{129}$$

where $w = \frac{x_4 + x_1}{2}$, $v = \frac{x_2 + x_3}{2}$, $\Delta w = x_4 - w$, $\Delta v = v - w$, and $\Delta u = x_3 - v$. Since Δu and

Δv are always less than Δw, as $\Delta w \to 0$, Equation (128) can be expanded in a series about $\Delta w=0$, $\Delta v = 0$, and $\Delta u =0$. Keeping terms to order five in $\Delta u, \Delta v, \Delta w$, or products of $\Delta u, \Delta v$, and Δw,

$$
\begin{aligned}
&M_n(w-\Delta w, w+\Delta v-\Delta u, w+\Delta v+\Delta u, w+\Delta w) \\
&\quad \approx \left[M_n(w) - 3M_{n+1}(w) + 3M_{n+2}(w) - M_{n+3}(w)\right]\frac{1}{6} \\
&\quad + \left[-M_n(w) + 4M_{n+1}(w) - 6M_{n+2}(w) + 4M_{n+3}(w) - M_{n+4}(w)\right]\frac{\Delta v}{12} \\
&\quad + [\, M_n(w) - 5M_{n+1}(w) + 10M_{n+2}(w) - 10M_{n+3}(w) \\
&\qquad + 5M_{n+4}(w) - M_{n+5}(w)\,]\frac{3\Delta v^2 + \Delta w^2 + \Delta u^2}{120} \\
&\quad + [\, -M_n(w) + 6M_{n+1}(w) - 15M_{n+2}(w) + 20M_{n+3}(w) - 15M_{n+4}(w) \\
&\qquad + 6M_{n+5}(w) - M_{n+6}(w)\,]\frac{2\Delta v^3 + \Delta v\Delta w^2 + 2\Delta u^2\Delta v}{180} \\
&\quad + [\, M_n(w) - 7M_{n+1}(w) + 21M_{n+2}(w) \\
&\qquad - 35M_{n+3}(w) + 35M_{n+4}(w) - 21M_{n+5}(w) + 7M_{n+6}(w) \\
&\qquad - M_{n+8}(w)\,]\frac{3\Delta v^2\Delta w^2 + \Delta u^2\Delta w^2 + 10\Delta u^2\Delta v^2}{5040} \\
&\quad + [\, -M_n(w) + 8M_{n+1}(w) - 28M_{n+2}(w) + 56M_{n+3}(w) \\
&\qquad - 70M_{n+4}(w) + 56M_{n+5}(w) - 28M_{n+6}(w) + 8M_{n+8}(w) \\
&\qquad - M_{n+9}(w)\,]\frac{\Delta v^3\Delta w^2 + \Delta u^2\Delta v\Delta w^2 + 5\Delta u^2\Delta v^3}{10080}.
\end{aligned}
\tag{129}
$$

For $\Delta w > 10^{-2}$, then functions are computed using recursion. For $\Delta w < 10^{-2}$, the expansion using up to fifth order terms must be used, and for $\Delta w < 10^{-3}$, the expansion using up to third order terms is used to evaluate the functions.

If the arguments are such that the four-argument function can be computed entirely by recursion to first kind functions, care must be taken to avoid unnecessary duplication of effort. Figure 50 shows that blindly recursing through the lower dimensional routines results in twice as many first kind evaluations as needed. The algorithm for computing four-argument functions tracks the variable separation at each step and ensures the minimal amount of computation.

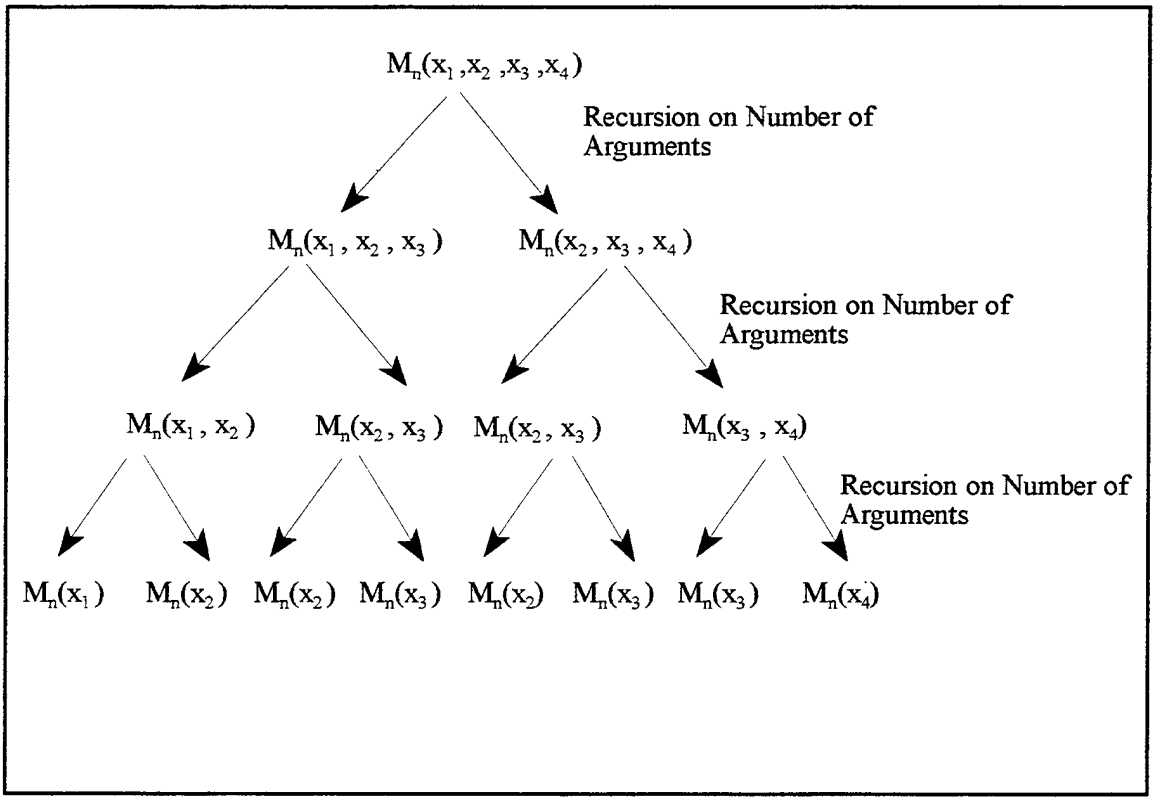

Figure 50. Number of single-argument moment functions that have to be computed to evaluate a four-argument function completely by recursion on number of arguments.

With this in mind, the algorithm for computing four-dimensional moment functions is:

Sort variables in ascending order
If $(x_4 - x_1) < 10^{-2}$ *then*
    Compute $M_n(x_1, x_2, x_3, x_4)$ *using expansion*
Else If $(x_3 - x_1) < 10^{-2}$ *and* $(x_4 - x_2) < 10^{-2}$ *then*
    Compute $M_n(x_1, x_2, x_3)$ *and* $M_n(x_2, x_3, x_4)$ *using expansion*
    Compute $M_n(x_1, x_2, x_3, x_4)$ *using recursion*
Else If $(x_3 - x_1) > 10^{-2}$ *and* $(x_4 - x_2) < 10^{-2}$ *then*
    If $(x_2 - x_1) < 10^{-2}$ *then*
        If $(x_3 - x_2) < 10^{-2}$ *then*
            Compute $M_n(x_2, x_3, x_4)$ *using expansion*
            Compute $M_n(x_1, x_2)$ *and* $M_n(x_2, x_3)$ *using expansion*
            Compute $M_n(x_1, x_2, x_3)$ *using recursion*
            Compute $M_n(x_1, x_2, x_3, x_4)$ *using recursion*
        Else If $(x_3 - x_2) > 10^{-2}$ *then*
            Compute $M_n(x_2, x_3, x_4)$ *using expansion*
            Compute $M_n(x_1, x_2)$ *using expansion*
            Compute $M_n(x_2)$ *and* $M_n(x_3)$
            Compute $M_n(x_2, x_3)$ *using recursion*
            Compute $M_n(x_1, x_2, x_3)$ *using recursion*
            Compute $M_n(x_1, x_2, x_3, x_4)$ *using recursion*
        End If
    Else If $(x_2 - x_1) > 10\text{-}2$ *then*
        If $(x_3 - x_2) < 10^{-2}$ *then*
            Compute $M_n(x_2, x_3, x_4)$ *using expansion*
            Compute $M_n(x_2, x_3)$ *using expansion*
            Compute $M_n(x_1)$ *and* $M_n(x_2)$
            Compute $M_n(x_1, x_2)$ *using recursion*

Compute $M_n(x_1, x_2, x_3)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

Else If $(x_3 - x_2) > 10^{-2}$ then

Compute $M_n(x_2, x_3, x_4)$ using expansion

Compute $M_n(x_1)$, $M_n(x_2)$, and $M_n(x_3)$

Compute $M_n(x_1, x_2)$ and $M_n(x_2, x_3)$ using recursion

Compute $M_n(x_1, x_2, x_3)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

End If

End If

Else If $(x_3 - x_1) < 10^{-2}$ and $(x_4 - x_2) > 10^{-2}$ then

If $(x_2 - x_1) < 10^{-2}$ then

If $(x_4 - x_3) < 10^{-2}$ then

Compute $M_n(x_1, x_2, x_3)$ using expansion

Compute $M_n(x_2, x_3)$ and $M_n(x_3, x_4)$ using expansion

Compute $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

Else If $(x_4 - x_3) > 10^{-2}$ then

Compute $M_n(x_1, x_2, x_3)$ using expansion

Compute $M_n(x_2, x_3)$ using expansion

Compute $M_n(x_3)$ and $M_n(x_4)$

Compute $M_n(x_3, x_4)$ using recursion

Compute $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

End If

Else If $(x_2 - x_1) > 10^{-2}$ then

If $(x_4 - x_3) < 10^{-2}$ then

Compute $M_n(x_1, x_2, x_3)$ using expansion

Compute $M_n(x_3, x_4)$ using expansion

Compute $M_n(x_2)$ and $M_n(x_3)$

Compute $M_n(x_2, x_3)$ using recursion

Compute $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

Else If $(x_4 - x_3) > 10^{-2}$ then

Compute $M_n(x_1, x_2, x_3)$ using expansion

Compute $M_n(x_2)$, $M_n(x_3)$, and $M_n(x_4)$

Compute $M_n(x_2, x_3)$ and $M_n(x_3, x_4)$ using recursion

Compute $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

End If

End If

Else If $(x_3-x_1) > 10^{-2}$ and $(x_4-x_2) > 10^{-2}$ then

If $(x_2 - x_1) < 10^{-2}$ then

If $(x_3-x_2) < 10^{-2}$ then

If $(x_4-x_3) < 10^{-2}$ then

Compute $M_n(x_1, x_2)$, $M_n(x_2, x_3)$, $M_n(x_3, x_4)$ using expansion

Compute $M_n(x_1, x_2, x_3)$ and $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

Else If $(x_4-x_3) > 10^{-2}$ then

Compute $M_n(x_1, x_2)$ and $M_n(x_2, x_3)$ using expansion

Compute $M_n(x_1, x_2, x_3)$ using expansion

Compute $M_n(x_3)$ and $M_n(x_4)$

Compute $M_n(x_3, x_4)$ using recursion

Compute $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

End If

Else If $(x_3-x_2) > 10^{-2}$ then

If $(x_4-x_3) < 10^{-2}$ then

Compute $M_n(x_2)$ and $M_n(x_3)$

Compute $M_n(x_2, x_3)$ using recursion

Compute $M_n(x_1, x_2)$ and $M_n(x_3, x_4)$ using expansions

Compute $M_n(x_1, x_2, x_3)$ and $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

Else If $(x_4-x_3) > 10^{-2}$ then

Compute $M_n(x_1, x_2)$ using function

Compute $M_n(x_2)$, $M_n(x_3)$, $M_n(x_4)$

Compute $M_n(x_2, x_3)$ and $M_n(x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3)$ and $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

End If

End If

Else If $(x_2 - x_1) > 10^{-2}$ then

If $(x_3-x_2) < 10^{-2}$ then

If $(x_4-x_3) < 10^{-2}$ then

Compute $M_n(x_2, x_3)$, $M_n(x_3, x_4)$ using expansion

Compute $M_n(x_1)$ and $M_n(x_2)$

Compute $M_n(x_1, x_2)$ using recursion

Compute $M_n(x_1, x_2, x_3)$ and $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

Else If $(x_4-x_3) > 10^{-2}$ then

Compute $M_n(x_2, x_3)$ using expansion

Compute $M_n(x_1)$, $M_n(x_2)$, $M_n(x_3)$, $M_n(x_4)$

Compute $M_n(x_1, x_2)$ and $M_n(x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3)$ and $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

End If

Else If $(x_3-x_2) > 10^{-2}$ then

If $(x_4-x_3) < 10^{-2}$ then

Compute $M_n(x_3, x_4)$ using expansion

Compute $M_n(x_1)$, $M_n(x_2)$, $M_n(x_3)$, $M_n(x_4)$

Compute $M_n(x_1, x_2)$ and $M_n(x_2, x_3)$ using recursion

Compute $M_n(x_1, x_2, x_3)$ and $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x_4)$ using recursion

Else If $(x_4-x_3) > 10^{-2}$ then

Compute $M_n(x_1)$, $M_n(x_2)$, $M_n(x_3)$, $M_n(x_4)$

Compute $M_n(x_1, x_2)$, $M_n(x_2, x_3)$, $M_n(x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3)$ and $M_n(x_2, x_3, x_4)$ using recursion

Compute $M_n(x_1, x_2, x_3, x4)$ using recursion

End If

End If

End If

End If.

Appendix B. Root Solving

B.1. Newton's Method

The equation

$$\mathbf{x}_{new} = \mathbf{x}_{old} - \mathbf{J}^{-1}(\mathbf{x}_{old})\,\mathbf{F}(\mathbf{x}_{old}) , \tag{130}$$

describes the Newton iteration for solving a nonlinear system of equations of the form

$$\mathbf{F}(\mathbf{x}) = \left[f_1(\mathbf{x}) , f_2(\mathbf{x}), \ldots , f_n(\mathbf{x})\right] = \mathbf{0} , \tag{131}$$

where

$$\mathbf{x} = \left(x_1 , x_2 , \ldots , x_n\right) . \tag{132}$$

The entries of the Jacobian matrix, **J**, are given as $J_{ij} = \frac{\partial F_i}{\partial x_j}$.

The method starts with some initial guess to the vector **x**. It then evaluates **F**(**x**) and $\mathbf{J}^{-1}(\mathbf{x})$ and uses these to compute a correction to the initial guess. This process is repeated until either the function values are zero to a given tolerance, or the change in the **x** values from the last iteration is less than the prescribed tolerance (Press, et. al. 1992: 372-375).

The method displays quadratic convergence, given an appropriate starting guess. If the initial guess is poor, the method may take many iterations to converge, or may not converge at all. The initial guess equations developed for the two- and three-dimensional

root solvers generate first guesses that generally lead to convergence to the desired tolerance in three to four iterations.

B.2. Calculation of the Input Flux Distribution Coefficients - 2D System

B.2.1. Derivation of 2D System Using Multi-Dimensional Moment Functions

As explained in Section 3.4, the coefficients of the exponential distribution must be generated from knowledge of the first two spatial moments of the distribution. The moments of the entering flux distribution are given by

$$\begin{aligned} \psi_A^{in} &= 2\alpha e^{X_{in}} M_0(X_{in}, Y_{in}) \\ \psi_{y'}^{in} &= \psi_A^{in} \left[h\rho(X_{in}, Y_{in}) \right] \\ \psi_{x'}^{in} &= \psi_A^{in} \left[\gamma b \rho(X_{in}, Y_{in}) + b\rho(X_{in} - Y_{in}, -Y_{in}) \right], \end{aligned} \tag{134}$$

where

$$\begin{aligned} X_{in} &= h\beta_y{}' + \gamma b \beta_x{}' \\ Y_{in} &= X_{in} - b\beta_x{}' . \end{aligned} \tag{134}$$

Given the equations for the flux moments, define

$$\rho_y = \frac{\psi_y^{in}}{h\,\psi_A^{in}} = \frac{M_1(X_{in}, Y_{in})}{M_0(X_{in}, Y_{in})} \tag{136}$$

and

$$\rho_x = \frac{\psi_x^{in}}{b\,\psi_A^{in}} = \gamma\,\rho_y + \frac{M_1(X_{in} - Y_{in}, -Y_{in})}{M_0(X_{in} - Y_{in}, -Y_{in})}. \tag{136}$$

Further, define

$$\rho_{xy} = \rho_x - \gamma\,\rho_y = \frac{M_1(X_{in} - Y_{in}, -Y_{in})}{M_0(X_{in} - Y_{in}, -Y_{in})}. \tag{137}$$

Equations (136) and (137) define a system of two equations in the two unknowns X_{in} and Y_{in}. Since the system is non-linear, the values of Xin and Yin must be found using Newton's Method on the system

$$f_1(X_{in}, Y_{in}) = \rho_y - \frac{M_1(X_{in}, Y_{in})}{M_0(X_{in}, Y_{in})} = 0, \tag{138}$$

and

$$f_2(X_{in}, Y_{in}) = \rho_{xy} - \frac{M_1(X_{in} - Y_{in}, -Y_{in})}{M_0(X_{in} - Y_{in}, -Y_{in})} = 0 \tag{139}$$

B.2.2. Note on Notation

The routines for doing the two-dimensional root solve were originally written for the triangle code, and the notation was slightly different. As shown in Figure 51, in the

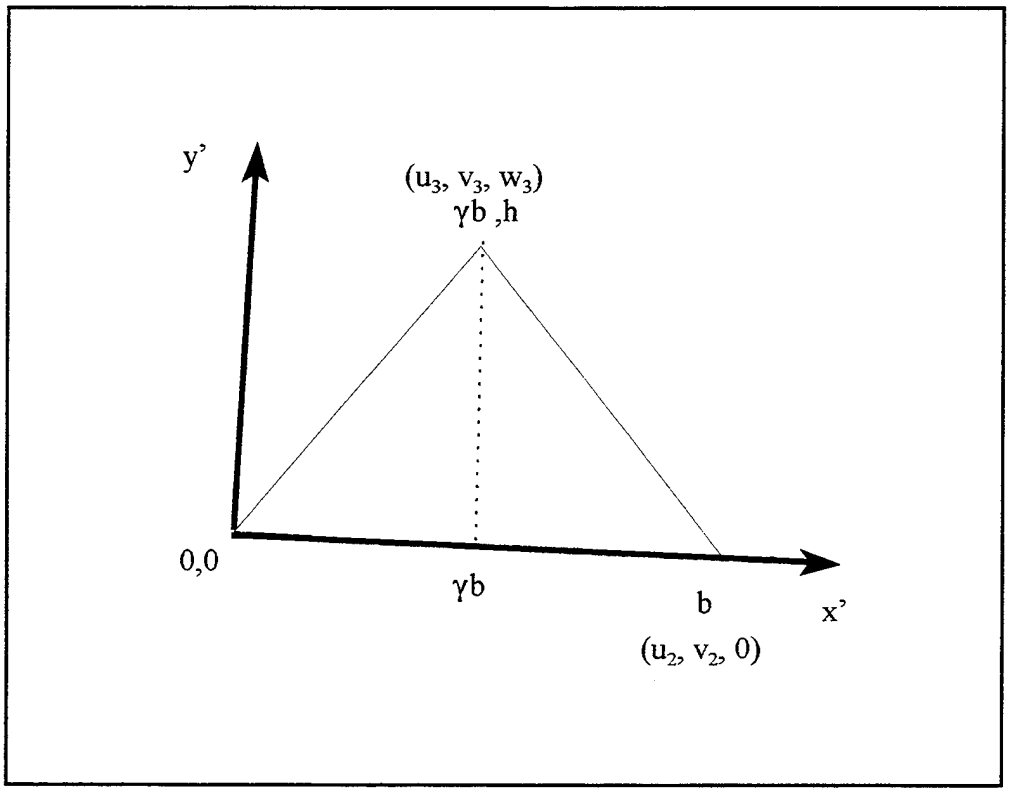

Figure 51. Entering Face Showing Node Coordinates in Both Local uvw and in Triangle Notation.

triangle system, apex points were labeled according the base length, b, the height, h, and γ. The Figure also shows the corresponding points in the (uvw) system. As explained earlier, on the entering face, u=u(v, w) so that rotation from the (uvw) to the (x',y') system is given by

$$x' = c_{1x}^{in}\, v + c_{2x}^{in}\, w$$
$$y' = c_{1y}^{in}\, w \,. \tag{140}$$

Then,

$$
\begin{aligned}
b &= c_{1x}^{in}\, v_2 \\
h &= c_{1y}^{in}\, w_3 \\
\gamma &= \frac{c_{1x}^{in}\, v_3 + c_{2x}^{in}\, w_3}{c_{1x}^{in}\, v_2},
\end{aligned}
\tag{141}
$$

and

$$
\begin{aligned}
X_{in} &= \beta_y c_{1y}^{in} w_3 + \beta_x (c_{1x}^{in} v_3 + c_{2x}^{in} w_3) \\
Y_{in} &= X_{in} - \beta_x c_{1x}^{in} v_2 .
\end{aligned}
\tag{142}
$$

Since it was carried over from the triangle code, the two-dimensional root solve routine is written in terms of b, h, and γ. This resulting system of equations is identical to Equations (138) and (139), so the discussion is valid for either choice of notation. The notation in terms of the uvw system is used to derive the initial guess equations.

B.2.3. Derivation of 2D System Using One-Dimensional Moments Functions

Applying the definition for recursion on number of arguments to Equations (138) and (139) gives

$$
f_1(X_{in}, Y_{in}) = \rho_y - \frac{M_1(Y_{in}) - M_1(X_{in})}{M_0(Y_{in}) - M_0(X_{in})} = 0 ,
\tag{143}
$$

and

$$f_2(X_{in}, Y_{in}) = \rho_{xy} - \frac{M_1(-Y_{in}) - M_1(X_{in} - Y_{in})}{M_0(-Y_{in}) - M_0(X_{in} - Y_{in})} = 0 \quad . \tag{144}$$

The functions f1 and f2 are now given completely in terms of single-argument functions. Further, the Jacobian can also be written in terms of single argument functions.

The number of two-dimensional root solves required per iteration for a given problem is just the average number of input faces per tetrahedron times the number of tetrahedron in the problem times the number of angles in the quadrature. In general, all tetrahedra are either case ±3 or 4, so that the average number of input faces is two. For an S8 quadrature, there are 80 angles, so that there are 160-times-the-number-of-tetrahedra two-dimensional root solves per iteration. For the system in the form of Equations (138) and (139), each Newton iteration requires 2 two-argument moment function evaluations and 4 three-argument moment function evaluations to compute the function and jacobian values. In the single argument form, a Newton iteration requires 4 single-argument moment function evaluations. This is much more computationally efficient than the multi-dimensional formulation. The equations are numerically well conditioned provided the values of X_{in} and Y_{in} are far enough apart in magnitude. Empirical testing shows that if $| X_{in} - Y_{in} | > 10^{-4}$ the single-argument formulation gives as much precision as the multi-dimensional equations.

B.2.4. Initial Guess Equations for the 2D System

Although Newton's method is quadratically convergent, if the initial guess is not sufficiently close to the root, the method may take a large number of iterations to converge, and may not even converge at all. The initial guess generator ensures that the two-dimensional root solving routine converges in a minimum number of iterations by generating an initial guess for X_{in} and Y_{in} that is close to the correct value. To understand how this works, we must first examine the ranges of ρ_y, ρ_{xy}, and their sum, ρ_{sum}. The equations defining spatial moments on tetrahedral faces are

$$\begin{pmatrix} \psi_A^{in} \\ \psi_w^{in} \\ \psi_v^{in} \end{pmatrix} = 2\int_0^{w_3} \frac{dw}{w_3} \int_{v_1(w)}^{v_2(w)} \frac{dv}{v_2} \begin{pmatrix} 1 \\ w \\ v \end{pmatrix} \psi^{in}(v,w) \,. \tag{146}$$

The equations over the face are manipulated such that

$$\int_0^{w_3} \frac{dw}{w_3} \int_{v_1(w)}^{v_2(w)} \frac{dv}{v_2} \psi^{in}(v,w) \Rightarrow \int_0^1 d\hat{w} \int_0^{\hat{w}} d\hat{v}\, \psi^{in}[v(\hat{v},\hat{w}), w(\hat{v},\hat{w})] \,, \tag{146}$$

where $w(\hat{v},\hat{w}) = (1-\hat{w})\, w_3$ and $v(\hat{v},\hat{w}) = v_3 - v_2\hat{v} + (v_2+v_3)\hat{w}$. Recall the definition of ρ_y:

$$\rho_y = \frac{\psi_y^{in}}{c_{1y}^{in} w_3 \psi_A^{in}} = \frac{\psi_w^{in}}{w_3 \psi_A^{in}} \,. \tag{148}$$

Using Equation (146),

$$\rho_y = \frac{2\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\,(1-\hat{w})\,\psi^{in}[v(\hat{v},\hat{w}),w(\hat{v},\hat{w})]}{2\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\,\psi^{in}[v(\hat{v},\hat{w}),w(\hat{v},\hat{w})]} \quad \therefore \quad 0<\rho_y<1\ . \tag{148}$$

Using Equations (137) and (140),

$$\rho_{xy} = \frac{\psi_v^{in}}{v_2\psi_A^{in}} - \frac{v_3}{v_2 w_3}\frac{\psi_w^{in}}{\psi_A^{in}}\ . \tag{149}$$

Again, using Equation (146),

$$\rho_{xy} = \frac{2\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\,(\hat{w}-\hat{v})\,\psi^{in}[v(\hat{v},\hat{w}),w(\hat{v},\hat{w})]}{2\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\,\psi^{in}[v(\hat{v},\hat{w}),w(\hat{v},\hat{w})]} \quad \therefore \quad 0<\rho_{xy}<1\ . \tag{150}$$

Finally, define $\rho_{sum} = \rho_y + \rho_{xy}$, and

$$\rho_{sum} = \frac{2\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\,(1-\hat{v})\,\psi^{in}[v(\hat{v},\hat{w}),w(\hat{v},\hat{w})]}{2\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\,\psi^{in}[v(\hat{v},\hat{w}),w(\hat{v},\hat{w})]} \quad \therefore \quad 0<\rho_{sum}<1\ . \tag{151}$$

Equations (148), (150), and (151) define the area in (ρ_x, ρ_{xy}) space which bounds all the values of ρ_x and ρ_{xy} that could satisfy the 2D system. This is shown in Figure 52.

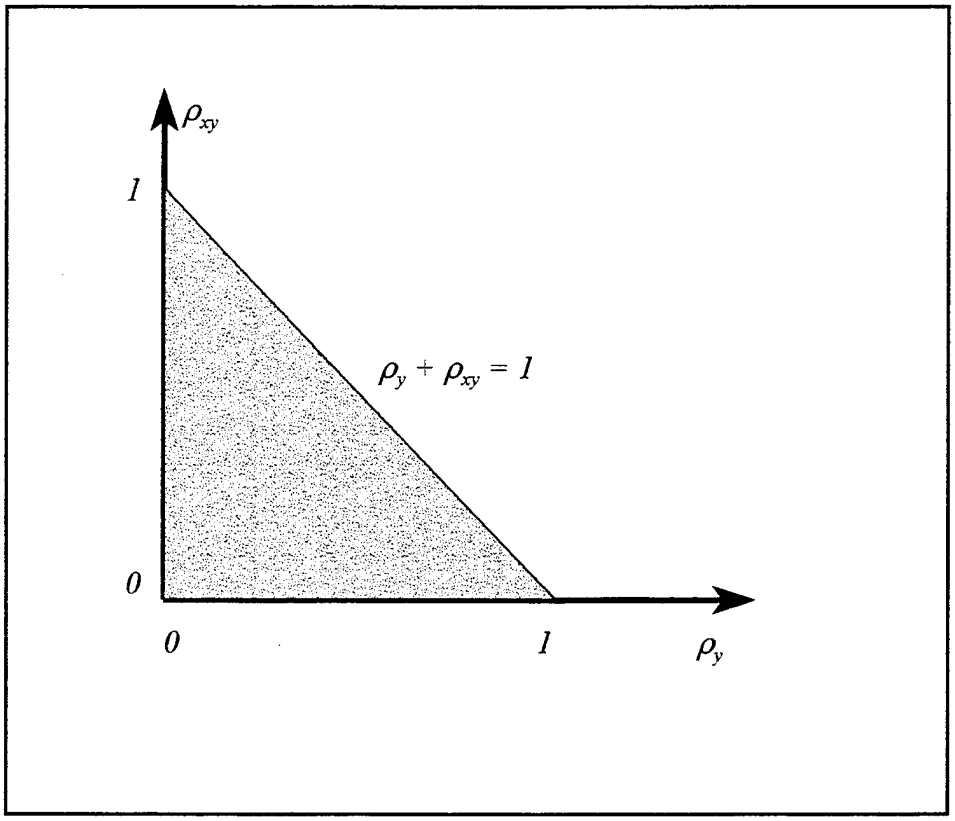

Figure 52. Area in (ρ_y, ρ_{xy}) Space Containing Valid Solutions to the 2D Rootsolve System.

Note that the area of possible solutions for Newton's method over a triangular face is itself a triangle.

The general form of the initial guess equations derives from an examination of the behavior of the solution as the points (ρ_y, ρ_{xy}) move toward the corners of the triangle. As both ρ_y and ρ_{xy} approach zero, the values of X_{in} and Y_{in} that satisfy Equations (138) and (139) are given by

$$X_{in} \approx -\frac{1}{\rho_y} \text{ , and ,}$$
$$Y_{in} \approx \frac{1}{\rho_{xy}} - \frac{1}{\rho_y} . \qquad (152)$$

If both ρ values are less than 0.06, the above equations give X_{in} and Y_{in} values that agree with the exact values to 12 digits, the current tolerance for the root-solved values. As the value of ρ_y approaches 1,

$$X_{in} \approx \frac{1}{1-\rho_{sum}} \text{ , and ,}$$
$$Y_{in} \approx \frac{1}{\rho_{xy}} . \qquad (153)$$

If the value of ρ_y is greater than 0.95, the above equations give both X and Y to 12 digits. For the case where ρ_{xy} approaches 1,

$$X_{in} \approx \frac{1}{1-\rho_{sum}} - \frac{1}{\rho_y} \text{ , and ,}$$
$$Y_{in} \approx -\frac{1}{\rho_y} . \qquad (154)$$

For values of ρ_{xy} greater than 0.95, the above equations give X_{in} and Y_{in} to 12 digits of accuracy. General case equations, that give good approximations over the entire area, can be interpolated from Equations (152), (153), and (154) by assuming those equations represent limiting cases. The general case equations are then

$$X_{in}^{initial} = \frac{(1-\rho_{sum}^2)^2}{1-\rho_{sum}} - \frac{(1-\rho_y^2)^2}{\rho_y}, \text{ and,}$$
$$Y_{in}^{initial} = \frac{(1-\rho_{xy}^2)^2}{\rho_{xy}} - \frac{(1-\rho_y^2)^2}{\rho_y}. \quad (155)$$

When the rootsolver is called, it computes the ρ values from the input flux moments. If the ρ's are such that Equations (152), (153), or (154) apply, those equations are used to generate X_{in} and Y_{in} without root-solving. Otherwise, Equations (154) are used to generate an initial guess for Newton's method. Using initial guesses generated by Equations (154), the root-solver is generally able to find X_{in} and Y_{in} values good to 10 digits using fewer than four iterations.

B.2.5. Computation of Distribution Coefficients from Root-Solved Values

Given the root-solved constants, β_x and β_y are found by inverting Equations (133):

$$\beta_x = \frac{X_{in} - Y_{in}}{b},$$
$$\beta_y = \frac{(1-\gamma)X_{in} + \gamma Y_{in}}{h}, \quad (156)$$

and

$$\alpha = \frac{\psi_A^{in}}{2e^{X_{in}} M_0(X_{in}, Y_{in})}. \quad (157)$$

Equations (156) work well regardless of the values of X_{in} and Y_{in}, the same is not true for Equation (157). For large negative values of X_{in} and Y_{in}, numerical problems caused by over and underflows can produce results that are not machine representable. For example, if X_{in} becomes a large negative number, regardless of the value of Y_{in}, the exponential term could evaluate as zero or an underflow, both of which cause the calculation to crash. In this cases, an alternate form of Equation (157) is used. For example, if $X_{in} = -750$ and $Y_{in} = 50$, then $e^{X_{in}} = 1.90\times10^{-326}$, an underflow, $M_0(X_{in}, Y_{in}) = 8.76\times10^{319}$, an overflow, but the product is $1.6666666666666667\text{x}10^{-6}$, a result that could be significant in the solution of the problem.

Using the definition of the exponential moments functions,

$$e^{X_{in}} M_0(X_{in}, Y_{in}) = \frac{-1}{X_{in}(Y_{in}-X_{in})} + \frac{e^{X_{in}}}{X_{in}(Y_{in}-X_{in})} - \frac{e^{X_{in}}}{Y_{in}(Y_{in}-X_{in})} + \frac{e^{X_{in}-Y_{in}}}{Y_{in}(Y_{in}-X_{in})}. \tag{158}$$

For large negative values of X_{in}, all but the first term on the right can be neglected, and Equation (158) can be approximated as

$$e^{X_{in}} M_0(X_{in}, Y_{in}) \approx \frac{-1}{X_{in}(Y_{in}-X_{in})}. \tag{159}$$

Empirical testing shows that the approximation can be used for $X_{in} < -30$ and $X_{in} - Y_{in} < -30$ with a maximum relative error of 10^{-12} (the tolerance used to compute the moments

functions). Using the example values of X_{in} = -750 and Y_{in} = 50, Equation (159) gives the correct result of $1.6666666666666667 \times 10^{-6}$, with the added benefit of not having to compute an exponential moment function. Therefore, Equation (159) is used whenever possible to compute α.

If the value of X_{in} is a large negative number and the conditions for using Equation (159) don't hold, Equation (157) can be modified using the identity

$$e^{X_{in}} M_0(X_{in}, Y_{in}) = e^{X_{in} - Y_{in}} M_0(X_{in} - Y_{in}, -Y_{in}) \,. \tag{160}$$

For example, if X_{in}=-750 and Y_{in}=-751, then $e^{-X_{in}} = 1.90 \times 10^{-326}$, an underflow, and $M_0(X_{in}, Y_{in}) = 1.20 \times 10^{323}$, an overflow. The product of the two is $2.28621637167 \times 10^{-3}$, a perfectly reasonable result. If we instead use Equation (160), $e^{X_{in} - Y_{in}} = 2.72$, $M_0(X_{in} - Y_{in}, -Y_{in}) = 8.41 \times 10^{-4}$, and the product of the two is $2.28621637167 \times 10^{-3}$, thereby avoiding overflow/underflow problems.

B.3. Calculation of the Source Distribution Coefficients: 3D System

B.3.1. Derivation of 3D System Using Multi-Dimensional Moment Functions

The assumed form of the source distribution for the EC method is

$$S(x,y,z) = ae^{bx + cy + dz} \tag{161}$$

For a given tetrahedron, the source moments are calculated using the scalar fluxes from the previous iteration. The source distribution coefficients are then chosen to match the zeroth and first source moments. Using the knowledge of the source distribution moments, the coefficients of the distribution can be computed by inverting Equations (84), giving a system of three equations in three unknowns which then has to be rootsolved. The equations are

$$\rho_w = \frac{S^w}{w_3 S^A} = \frac{M_1(X_s,Y_s,Z_s)}{M_0(X_s,Y_s,Z_s)} \tag{162}$$

$$\rho_v = \frac{S^v}{v_2 S^A} = \frac{v_3}{v_2}\rho_w + \frac{M_1(Y_s-X_s,Z_s-X_s,-X_s)}{M_0(Y_s-X_s,Z_s-X_s,-X_s)} \tag{163}$$

$$\rho_u = \frac{S^u}{u_1 S^A} = \left[\frac{u_3 v_2 - u_2 v_3}{u_1 v_2}\right]\rho_w + \frac{u_2}{u_1}\rho_v + \frac{M_1(Y_s-Z_s,X_s-Z_s,-Z_s)}{M_0(Y_s-Z_s,X_s-Z_s,-Z_s)} \tag{164}$$

Now define

$$\rho_{vw} = \rho_v - \frac{v_3}{v_2}\rho_w = \frac{M_1(Y_s - X_s, Z_s - X_s, -X_s)}{M_0(Y_s - X_s, Z_s - X_s, -X_s)} \tag{165}$$

$$\rho_{uvw} = \rho_u - \frac{u_2}{u_1}\rho_v - \left[\frac{u_3 v_2 - u_2 v_3}{u_1 v_2}\right]\rho_w = \frac{M_1(Y_s - Z_s, X_s - Z_s, -Z_s)}{M_0(Y_s - Z_s, X_s - Z_s, -Z_s)} \tag{166}$$

giving the system of equations

$$f_1(X_s, Y_s, Z_s) = \rho_w - \frac{M_1(X_s, Y_s, Z_s)}{M_0(X_s, Y_s, Z_s)} = 0 \tag{167}$$

$$f_2(X_s, Y_s, Z_s) = \rho_{vw} - \frac{M_1(Y_s - X_s, Z_s - X_s, -X_s)}{M_0(Y_s - X_s, Z_s - X_s, -X_s)} = 0 \tag{168}$$

and

$$f_3(X_s, Y_s, Z_s) = \rho_{uvw} - \frac{M_1(Y_s - Z_s, X_s - Z_s, -Z_s)}{M_0(Y_s - Z_s, X_s - Z_s, -Z_s)} = 0 \tag{169}$$

The above system contains three equations in the three unknowns X_s, Y_s, and Z_s.

B.3.2. Derivation of 3D System Using One-Dimensional Moment Functions

Just as in the two-dimensional case, the three-dimensional system can be written entirely in terms of single-argument function using recursion on number of arguments. This leads to the following system of equations:

$$f_1(X_s,Y_s,Z_s) = \rho_w - \frac{(Y_s-Z_s)M_1(X_s) + (Z_s-X_s)M_1(Y_s) + (X_s-Y_s)M_1(Z_s)}{(Y_s-Z_s)M_0(X_s) + (Z_s-X_s)M_0(Y_s) + (X_s-Y_s)M_0(Z_s)} = 0 , \quad (170)$$

$$f_2(X_s,Y_s,Z_s) = \rho_{vw} - \frac{(Z_s-Y_s)M_1(-X_s) - Z_sM_1(Y_s-X_s) + Y_sM_1(Z_s-X_s)}{(Z_s-Y_s)M_0(-X_s) - Z_sM_0(Y_s-X_s) + Y_sM_0(Z_s-X_s)} = 0 , \quad (171)$$

and

$$f_3(X_s,Y_s,Z_s) = \rho_{uvw} - \frac{Y_sM_1(X_s-Z_s) - X_sM_1(Y_s-Z_s) + (X_s-Y_s)M_1(-Z_s)}{Y_sM_0(X_s-Z_s) - X_sM_0(Y_s-Z_s) + (X_s-Y_s)M_0(-Z_s)} = 0 . \quad (172)$$

Now both the functions and their derivatives used in the Jacobian are given completely in terms of single-argument moment functions. One Newton iteration using the multi-dimensional form of the system requires three calls to the three-dimensional routines and nine calls to the four-dimensional routines. In the single-argument form, a Newton iteration requires only nine calls to the single-argument routines. As in the 2D case, the above system is numerically ill conditioned as any of the variables X_s, Y_s, Z_s, approach zero, or as the difference between any two variables approaches zero. The system in

terms of multiple-argument functions agrees with the multi-dimensional formulation when any of the arguments has magnitude less than 10^{-4}, or when the difference between any two arguments is less than 10^{-4} . Since it is much more efficient, the system in terms of single-argument functions is used whenever possible.

B.3.3. Initial Guess Equations for the 3D System

The similarity between the two- and three-dimensional systems suggests that the initial guess equations would share some of the same properties. This is indeed the case. The equations for generating an initial guess are found using the same procedure as the 2D case, beginning with an examination of the ranges of ρ_w, ρ_{vw}, and ρ_{uvw}. The equations defining the source moments over the tetrahedron volume are

$$\begin{pmatrix} S_A \\ S_u \\ S_v \\ S_w \end{pmatrix} = 6\int_0^{w_3} \frac{dw}{w_3} \int_{v_1(w)}^{v_2(w)} \frac{dv}{v_2} \int_{u_{in}(v,w)}^{u_{out}(v,w)} \frac{du}{u_1} \begin{pmatrix} 1 \\ u \\ v \\ w \end{pmatrix} S(u,v,w) \; . \tag{173}$$

Using several changes of variables,

$$\begin{aligned} & 6\int_0^{w_3} \frac{dw}{w_3} \int_{v_1(w)}^{v_2(w)} \frac{dv}{v_2} \int_{u_{in}(v,w)}^{u_{out}(v,w)} \frac{du}{u_1} \; S(u,v,w) \\ & \qquad \Rightarrow 6\int_0^1 d\hat{w} \int_0^{\hat{w}} d\hat{v} \int_0^{\hat{v}} d\hat{u} \; S\left[u(\hat{u},\hat{v},\hat{w}), v(\hat{u},\hat{v},\hat{w}), w(\hat{u},\hat{v},\hat{w})\right] , \end{aligned} \tag{174}$$

where

$$
\begin{aligned}
w(\hat{u},\hat{v},\hat{w}) &= (1-\hat{w})w_3 \\
v(\hat{u},\hat{v},\hat{w}) &= v_3 - v_2\hat{v} + (v_2+v_3)\hat{w} \\
u(\hat{u},\hat{v},\hat{w}) &= u_3 + u_1\hat{u} - u_2\hat{v} + (u_2-u_3)\hat{w} \ .
\end{aligned} \tag{175}
$$

From Equation (162),

$$
\rho_w = \frac{S^w}{w_3 S^A} = \frac{6\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\int_0^{\hat{v}} d\hat{u}\ (1-\hat{w})\ S(\hat{u},\hat{v},\hat{w})}{6\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\int_0^{\hat{v}} d\hat{u}\ S(\hat{u},\hat{v},\hat{w})} \ \therefore\ 0<\rho_w<1\ . \tag{176}
$$

From Equation (165),

$$
\rho_{vw} = \rho_v - \frac{v_3}{v_2}\rho_w = \frac{6\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\int_0^{\hat{v}} d\hat{u}\ (\hat{v}-\hat{w})\ S(\hat{u},\hat{v},\hat{w})}{6\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\int_0^{\hat{v}} d\hat{u}\ S(\hat{u},\hat{v},\hat{w})} \ \therefore\ 0<\rho_{vw}<1\ . \tag{177}
$$

From Equation (166),

$$
\rho_{uvw} = \rho_u - \frac{u_2}{u_1}\rho_v - \left[\frac{u_3 v_2 - u_2 v_3}{u_1 v_2}\right]\rho_w = \frac{6\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\int_0^{\hat{v}} d\hat{u}\ \hat{u}\ S(\hat{u},\hat{v},\hat{w})}{6\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\int_0^{\hat{v}} d\hat{u}\ S(\hat{u},\hat{v},\hat{w})} \ \therefore\ 0<\rho_{uvw}<1. \tag{178}
$$

Finally, the sum of the ρ's,

$$\rho_{sum} = \rho_w + \rho_{vw} + \rho_{uvw} = \frac{6\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\int_0^{\hat{v}} d\hat{u}\ (1-\hat{v}+\hat{u})\,S(\hat{u},\hat{v},\hat{w})}{6\int_0^1 d\hat{w}\int_0^{\hat{w}} d\hat{v}\int_0^{\hat{v}} d\hat{u}\ S(\hat{u},\hat{v},\hat{w})} \quad \therefore\ 0<\rho_{sum}<1\ . \qquad (179)$$

These equations define the volume in ρ_w, ρ_{vw}, ρ_{uvw} space, shown in Figure 53, that contains all values (ρ_w, ρ_{vw}, ρ_{uvw}) that satisfy the system. Note that in analogy with the two-dimensional case, the volume of attainable values for the ρ's of the source distribution in a tetrahedral cell is itself a tetrahedron.

The general equation for initial values is found by examining the behavior of the system as the points (ρ_w, ρ_{vw}, ρ_{uvw}) move to the tetrahedron apexes. As the ρ's approach zero, the following equations hold

$$X_s \approx \frac{1}{\rho_{vw}} - \frac{1}{\rho_w},$$

$$Y_s \approx \frac{-1}{\rho_w}, \qquad (181)$$

$$\text{and} \quad Z_s \approx \frac{1}{\rho_{uvw}} - \frac{1}{\rho_w}.$$

For ρ_w, ρ_{vw} and ρ_{uvw} all less than 4.0×10^{-2}, the Equations (180) give X_s, Y_s, and Z_s to twelve digits, and rootsolving is not necessary. As ρ_w approaches 1,

$$X_s \approx \frac{1}{\rho_{vw}},$$

$$Y_s \approx \frac{1}{1-\rho_{sum}}, \qquad (181)$$

$$\text{and}, \quad Z_s \approx \frac{1}{\rho_{uvw}}.$$

For $\rho_w > 0.95$ and $\rho_{sum} > 0.99$, Equations (181) gives X_s, Y_s, and Z_s to twelve digits, and rootsolving is not necessary. As ρ_{vw} approaches 1,

$$X_s \approx -\frac{1}{\rho_w},$$

$$Y_s \approx \frac{1}{1-\rho_{sum}} - \frac{1}{\rho_w}, \qquad (182)$$

$$\text{and}, \quad Z_s \approx \frac{1}{\rho_{uvw}} - \frac{1}{\rho_w}.$$

For $\rho_{vw} > 0.95$ and $\rho_{sum} > 0.99$, Equations (182) gives X_s, Y_s, and Z_s to twelve digits, and rootsolving is not necessary. As ρ_{uvw} approaches 1,

$$X_s \approx \frac{1}{\rho_{vw}} - \frac{1}{\rho_w},$$

$$Y_s \approx \frac{1}{1-\rho_{sum}} - \frac{1}{\rho_w}, \qquad (183)$$

$$\text{and}, \quad Z_s \approx \frac{1}{\rho_w}.$$

For $\rho_{uvw} > 0.95$ and $\rho_{sum} > 0.99$, Equations (183) gives X_s, Y_s, and Z_s to twelve digits, and rootsolving is not necessary. Assuming that the behaviors in the corners are limiting

cases of a more general function, a global set of initial guess equations can be inferred:

$$X_s^{initial} = \frac{(1-\rho_{vw}^2)^2}{\rho_{vw}} - \frac{(1-\rho_w^2)^2}{\rho_w},$$

$$Y_s^{initial} = \frac{(1-(1-\rho_{sum}^2))^2}{1-\rho_{sum}} - \frac{(1-\rho_w^2)^2}{\rho_w}, \text{ and }, \tag{184}$$

$$Z_s^{initial} = \frac{(1-\rho_{uvw}^2)^2}{\rho_{uvw}} - \frac{(1-\rho_w^2)^2}{\rho_w}.$$

The initial values generated by Equations (184) generally allow the rootsolver to converge in fewer than four iterations.

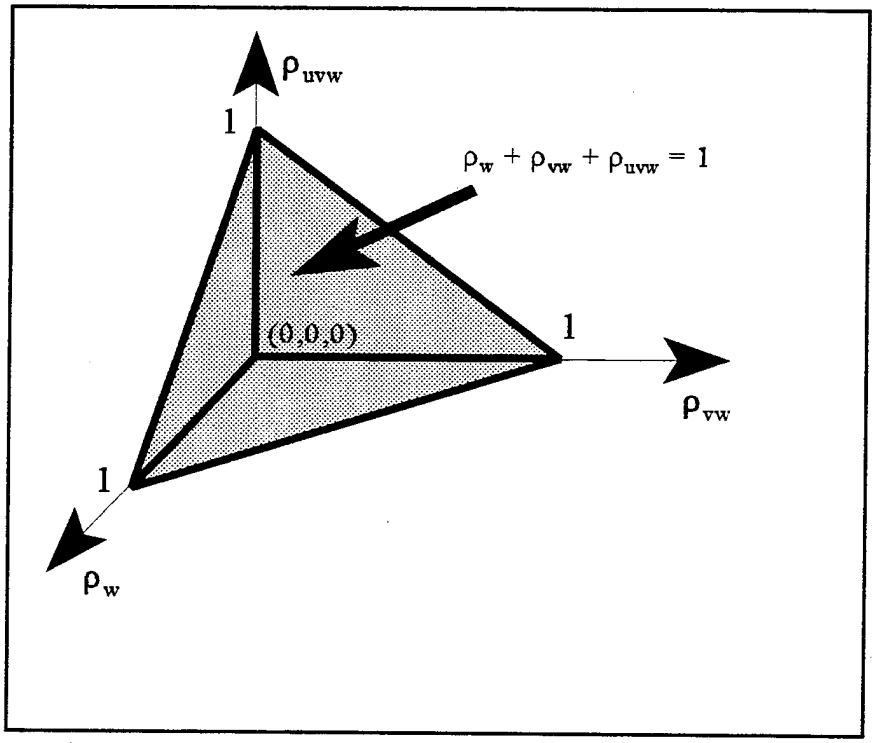

Figure 53. Phase Space Volume Containing Solutions to 3D RootSolving System.

B.3.4. Computation of Distribution Coefficients from Rootsolved Values

Given the root-solved values of X_s, Y_s, and Z_s, the values of the exponential distribution coefficients are found by inverting Equations (84), giving,

$$b = \frac{Y_s - Z_s}{u_1}$$

$$c = \frac{-u_1 X_s + (u_1 - u_2) Y_s + u_2 Z_s}{u_1 v_2} \tag{185}$$

$$d = \frac{u_1 v_3 X_s + (u_1 v_2 - u_3 v_2 - u_1 v_3 + u_2 v_3) Y_s + (u_3 v_2 - u_2 v_3) Z_s}{u_1 v_2 w_3},$$

and,

$$a = \frac{S^A}{6 e^{Y_s} M_0(X_s, Y_s, Z_s)}. \tag{186}$$

For extreme values of X_s, Y_s, and Z_s, Equation (186) is inadequate. If Y_s is a large negative number, the exponential term could be a very small number, while the moments function is a very large number. Although when multiplied together the product of the two is reasonable, the machine may assign the exponential term either a value of zero, or underflow, either of which would produce a non-machine-representable result. To avoid this, alternate forms of Equation (186) are used based on the values of X_s, Y_s, and Z_s. Applying the definition of the exponential moment functions and expanding,

$$e^{Y_s}M_0(X_s,Y_s,Z_s) = \frac{e^{Y_s} - e^{Y_s-X_s}}{X_s(Y_s-X_s)(Z_s-X_s)} + \frac{1-e^{Y_s}}{Y_s(Y_s-X_s)(Z_s-X_s)} + \frac{1-e^{Y_s}}{Y_s(Z_s-Y_s)(Z_s-X_s)} + \frac{e^{Y_s} - e^{Y_s-Z_s}}{Z_s(Z_s-Y_s)(Z_s-X_s)}. \quad (187)$$

If $Y_s < -30$, $Y_s - X_s < -32$, and $Y_s - Z_s < -32$, then Equation (187) can be approximated to good precision using

$$e^{Y_s}M_0(X_s,Y_s,Z_s) \approx \frac{1}{Y_s(Y_s-X_s)(Z_s-X_s)} + \frac{1}{Y_s(Z_s-Y_s)(Z_s-X_s)} = \frac{1}{Y_s(Y_s-X_s)(Z_s-Y_s)}. \quad (188)$$

For example, if Ys= -750, Xs = -50 and Zs = -51, then $e^{Y_s} = 1.901\times10^{-309}$, which could be interpreted as either zero, or an underflow, either of which blows up Equation (186). However, for these values, $e^{Y_s}M_0(X_s,Y_s,Z_s) = 2.72498126\times10^{-9}$. Using the approximation, Equation (188), gives the correct result, $2.72498126\text{x}10^{-9}$, but avoids non-numerical results. Further, Equation (188) doesn't require the computation of a three-argument moment function, which is much faster. Therefore, Equation (188) is used instead of Equation (186) whenever possible.

For situations where all the arguments of the moments function in Equation (186) are much less than zero, but the conditions for using Equation (188) don't hold, the identity

$$e^{Y_s}M_0(X_s,Y_s,Z_s) = e^{Y_s-X_s}M_0(Y_s-X_s,Z_s-X_s,-X_s) \quad (189)$$

can be used. For example, if X_s = -1001, Y_s = -1000, and Z_s = -1002, $e^{Y_s} = 5.075\times10^{-435}$, and $M_0(X_s, Y_s, Z_s) = 2.899\times10^{431}$, but the product of the two is 0.00147158747978. Equation (189) becomes $e^{1}M_0(1,2,1001) = 0.00147158747978$, which gives the correct result without producing overflows or underflows.

Appendix C. Tetrahedral Splitting

The splitting of arbitrarily oriented tetrahedra along the streaming direction can be broken down into two steps. The first step is case-dependent and consists of determining the identification of the nodes (vertices) defining each sub-tetrahedron. The second step is case-independent and consists of constructing the rotation and translation matrices needed to rotate/translate the node coordinates to the local sub-tetrahedron coordinate system, computing the subcell coordinates in the subcell local coordinate system, and computing the sub-tetrahedron entering and exiting face areas and volume. Since the splitting routines are designed to support the second step, this section begins by discussing the case-independent second step first, and then examines the details of determining the rotation matrices and sub-tetrahedron node coordinates for each of the four cases.

C.1 Construction of the Local Sub-Tetrahedron Coordinate System

The splitting routines identify sub-tetrahedra that are arbitrarily oriented with respect to the centroid coordinate system, an example of which is shown on the left-hand side of Figure 54. Application of the quadrature requires the construction of a local sub-tetrahedron coordinate system as shown in the right-hand side of the figure. For each sub-tetrahedron, the splitting routine generates a node matrix, $\mathbf{T}_{xyz}$, that contains the centroid-system coordinates of nodes defining the tetrahedron,

$$\mathbf{T}_{xyz} = \begin{pmatrix} A_x & B_x & C_x & D_x \\ A_y & B_y & C_y & D_y \\ A_z & B_z & C_z & D_z \end{pmatrix}. \tag{190}$$

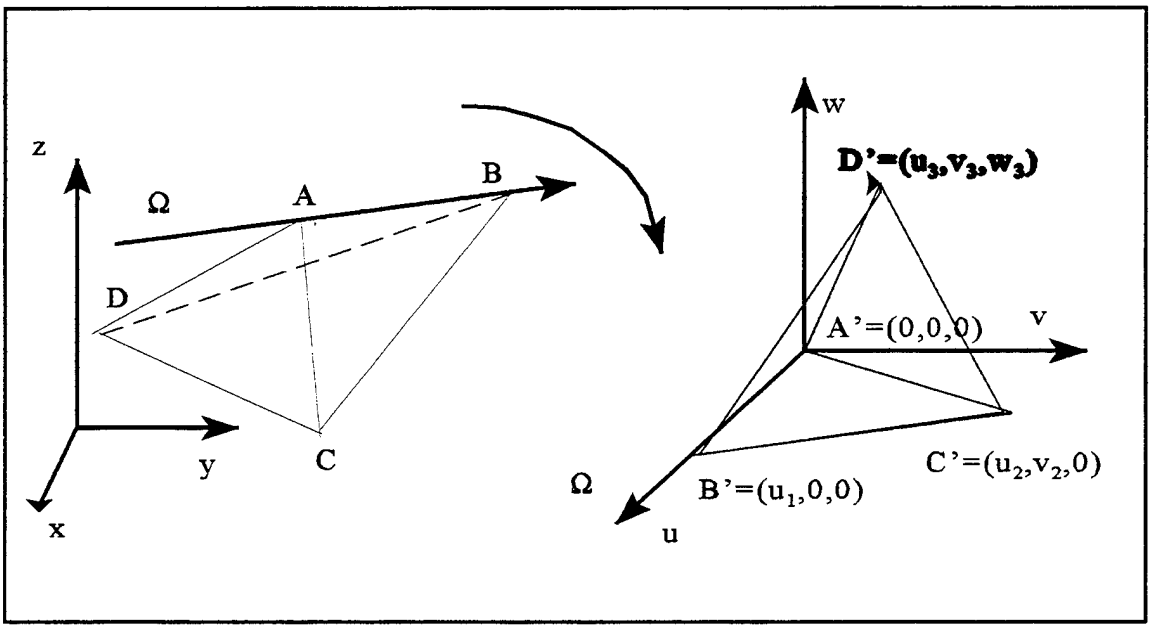

Figure 54. Case 1 Tetrahedron in Global xyz Coordinate System (left figure) and in Local uvw Coordinate System (right figure)

The nodes are specifically ordered so that when transformed, the local coordinate matrix has the form

$$\mathbf{T}_{uvw} = \begin{pmatrix} 0 & u_1 & u_2 & u_3 \\ 0 & 0 & v_2 & v_3 \\ 0 & 0 & 0 & w_3 \end{pmatrix}. \tag{191}$$

The transformation is accomplished using

$$\mathbf{T}_{uvw} = \mathbf{R}\,\mathbf{T}_{xyz} - \mathbf{t} \tag{192}$$

where **R**, the rotation matrix from the centroid coordinate system to the local uvw coordinate system, is defined as

$$\mathbf{R} = \begin{pmatrix} \hat{e}_u \cdot \hat{e}_x & \hat{e}_v \cdot \hat{e}_x & \hat{e}_w \cdot \hat{e}_x \\ \hat{e}_u \cdot \hat{e}_y & \hat{e}_v \cdot \hat{e}_y & \hat{e}_w \cdot \hat{e}_y \\ \hat{e}_u \cdot \hat{e}_z & \hat{e}_v \cdot \hat{e}_z & \hat{e}_w \cdot \hat{e}_z \end{pmatrix}, \tag{193}$$

and the translation vector

$$\mathbf{t} = \mathbf{R} \begin{pmatrix} A_x \\ A_y \\ A_z \end{pmatrix} \tag{194}$$

is just the origin of the local coordinate system in rotated coordinates. (Note that the translation is accomplished after the rotation because for case 4 tetrahedra the translation vector is unknown until after the rotation). The rotation matrix is defined in terms of dot products to avoid repeated computation of sines and cosines.

The rotation matrix is composed of dot products between the unit vectors along the coordinate axes of each system. Since the centroid coordinate system is simply the global coordinate system translated to the centroid, the unit vectors are the same as those of the global system. In the local uvw system, the u-axis is defined by the streaming direction. The unit vector in the u-direction is simply

$$\hat{e}_u = \hat{\Omega} = \mu \hat{e}_x + \eta \hat{e}_y + \xi \hat{e}_z \,. \tag{195}$$

Referring to Figure 54, since the point C lies in the uv-plane, the vector $\vec{P}_{AC}$ must also lie in the uv-plane and the unit vector in the w-direction can be defined as

$$\hat{e}_w = \hat{e}_u \times \frac{\vec{P}_{AC}}{|\vec{P}_{AC}|}, \tag{196}$$

then

$$\hat{e}_v = \hat{e}_w \times \hat{e}_u . \tag{197}$$

R and **t** are also used to transform the source distribution coefficients to the local system, and to transform the cell flux moments back to the centroid system.

C.2 Case 4 Tetrahedra Splitting

Case 4 tetrahedra are characterized by having two input and two output faces. When split along the streaming direction, the case 4 tetrahedron produces four case 1 tetrahedra as shown in Figure 55.

In essence, the problem of splitting a Case 4 tetrahedron reduces to finding the coordinates of the points P_5 and P_6 in Figure 56. Given these points, the matrices containing the nodes and the rotation and translation matrices can be generated. There are several ways to approach the problem of finding points P_5 and P_6. One approach is to note the following:

1) Point P_5 is on the line $\overline{P_1P_3}$
2) Point P_6 is on the line $\overline{P_2P_4}$

3) Point P_5 is on the line parallel to Ω passing through P_6

and solve the resulting system of six equations for the coordinates of P_5 and P_6. Another approach is to note:

1) Point P_5 lies in the plane defined by $P_1P_2P_3$ and the plane defined by $P_1P_3P_4$

2) Point P_6 lies in the plane defined by $P_1P_2P_4$ and the plane defined by $P_2P_3P_4$

3) Point P5 is on the line parallel to Ω passing through Point P_6.

and solve the resulting set of equations for the P_5 and P_6 coordinates. Both of these approaches were investigated, in each case the resulting equations for the node coordinates were very large and unwieldy, and did not always work.

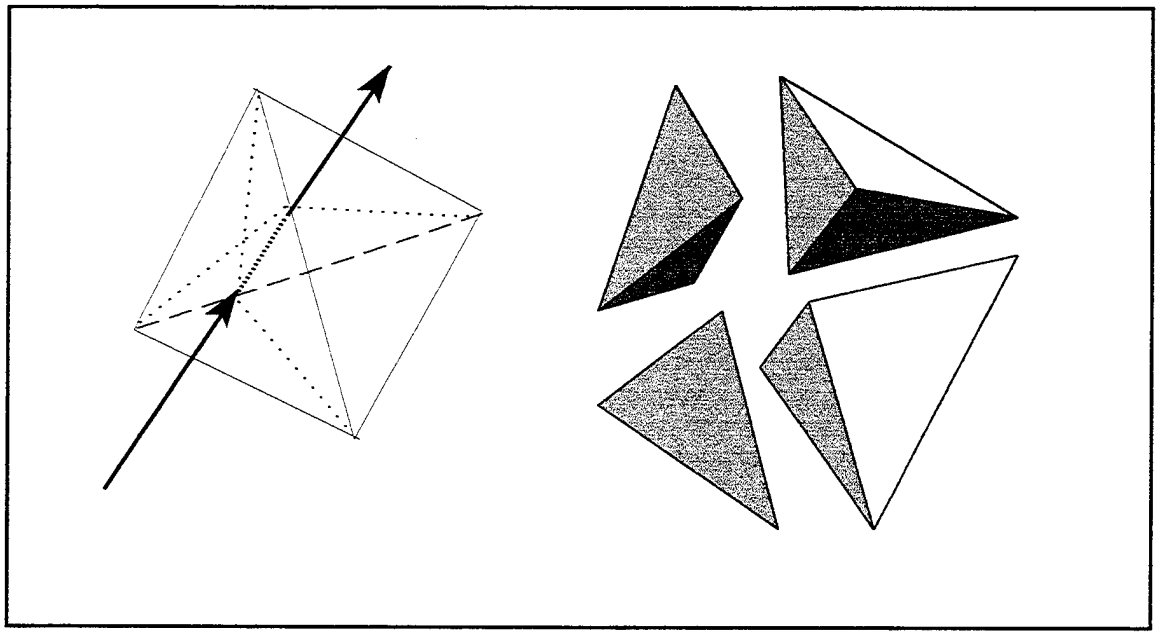

Figure 55. Case 4 Tetrahedron and Resulting Sub-Tetrahedra.

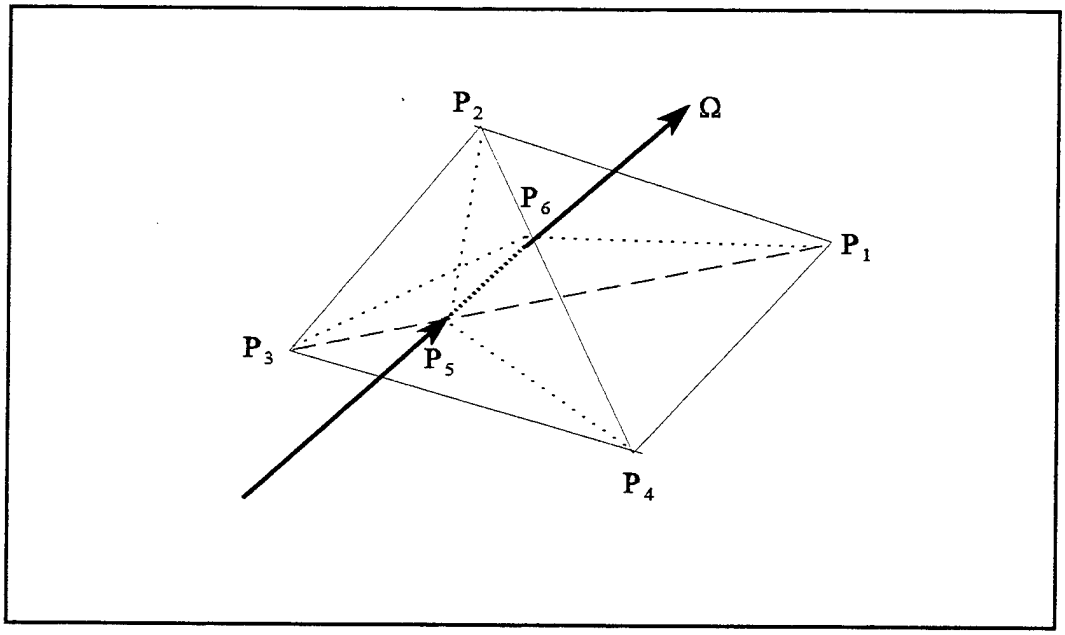

Figure 56. Case 4 Tetrahedron Showing Point Labeling Convention (Configuration 1).

The approach that finally did work relies on the fact that since the translation, Equation (194), is defined in terms of rotated coordinates, then it is only necessary to know P_5 and P_6 in the rotated coordinates, and the rotation can be done first. Once rotated, the problem of finding the intersection of three lines, $\overline{P_1P_3}$, $\overline{P_2P_4}$ and a line parallel to the streaming direction, in three dimensions can be reduced to finding the intersection of two lines in a plane.

The first step is to label the tetrahedron nodes. When passed to the splitting routine, the coordinates of the parent tetrahedron nodes are known, but we have no idea how the nodes are oriented with respect to one another. The nodes are labeled using the following convention, illustrated in Figure 56. The point where the streaming vector

enters the tetrahedron is labeled P_5, and the exiting point, P_6. The nodes defining the edge containing P_5 (the edge common to the two entering faces) are labeled P_1 and P_3. The nodes defining the edge containing P_6 (the edge common to the two exiting faces) are labeled P_2 and P_4. There are two possible point configurations which can result: the one shown in Figure 56, and one similar to Figure 56 in which the points P_1 and P_3 are exchanged. (Another approach is simply to swap labels P_1 and P_3, the method described is slightly more efficient and straightforward). The routine first checks the orientation and uses a set of equations consistent with the orientation. For configuration 1, shown in Figure 56, the sub-tetrahedra are: $\mathbf{T}_1$=(P_5,P_6,P_1,P_2), $\mathbf{T}_2$=(P_5,P_6,P_2,P_3), $\mathbf{T}_3$=(P_5,P_6,P_3,P_4), and $\mathbf{T}_4$=(P_5,P_6,P_4,P_1). And for configuration 2 the sub-tetrahedra are: $\mathbf{T}_1$=(P_5,P_6,P_2,P_1), $\mathbf{T}_2$=(P_5,P_6,P_3,P_2), $\mathbf{T}_3$=(P_5,P_6,P_4,P_3), and $\mathbf{T}_4$=(P_5,P_6,P_1,P_4). Again, we note that the node ordering is important in identifying the positions of the nodes once transformed into the subcell system.

C.2.1 Sub-Tetrahedron 1 (Configuration 1)

Before transforming coordinates, we recognize that sub-tetrahedron 1 is defined as

$$\mathbf{T}_1 = \begin{pmatrix} x_5 & x_6 & x_1 & x_2 \\ y_5 & y_6 & y_1 & y_2 \\ z_5 & z_6 & z_1 & z_2 \end{pmatrix}, \tag{198}$$

where the node coordinates are in the centroid coordinate system. The node coordinates are given in the centroid coordinate system because the resulting rotation and translation

matrices will also be used to transform the source moments from the centroid system and to transform the subcell moments back into the centroid system. A typical example is shown in Figure 57, where the xyz axes represent the centroid coordinate system, the u'v'w' axes represent the rotated (but not yet translated) coordinate system, and the uvw axes the fully transformed, local sub-tetrahedron coordinate system.

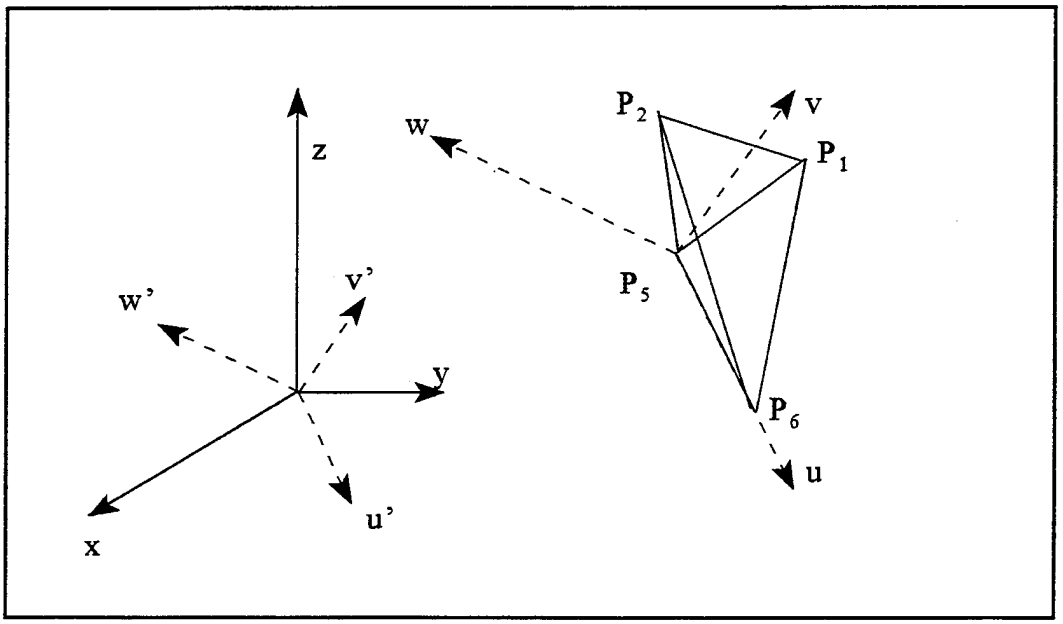

Figure 57. Case 4, Sub-Tetrahedron 1 in Relation to Global, Rotated, and Local Sub-Tetrahedron Coordinate Systems.

For sub-tetrahedron 1 (in fact for all four sub-tetrahedra), by construction:

$$\hat{e}_u = \hat{\Omega} = \mu \hat{e}_x + \eta \hat{e}_y + \xi \hat{e}_z \,. \tag{199}$$

Further, referring to Figure 58, P_1 and P_3 both lie in the plane perpendicular to $\hat{e}_w$, thus

$$\hat{e}_w = \hat{\Omega} \times \frac{\vec{P}_{31}}{|\vec{P}_{31}|} \tag{200}$$

where $\vec{P}_{31}$ is the vector from point P_3 to point P_1 and

$$\hat{e}_v = \hat{e}_w \times \hat{e}_u \,. \tag{201}$$

(Note that $\hat{e}_u = \hat{e}_{u'}$, $\hat{e}_v = \hat{e}_{v'}$, and $\hat{e}_w = \hat{e}_{w'}$.) We can generate the rotation matrix using the unit vectors in Equation (193) and rotate the *entire* parent tetrahedron into the u'v'w' system, as shown in Figure 58. The node coordinates are now given in the rotated coordinate system.

Next, examine the projection of the edges into the v'w'-plane, shown in Figure 59. From the line connecting P'_1 and P'_3, it is obvious that,

$$w'_5 = w'_6 = w'_1 = w'_3 \,. \tag{202}$$

The equation of the line from P'_2 to P'_4 is just

$$w' = \frac{(w'_2 - w'_4)}{(v'_2 - v'_4)}(v' - v'_4) + w'_4 \,. \tag{203}$$

Using Equation (202) and applying Equation (203) at the point P'_5 (i.e. let $w' = w'_5$, then $v' = v'_5$, and solve for v'_5)

$$v'_5 = \frac{(w'_1 - w'_4)}{(w'_2 - w'_4)}(v'_2 - v'_4) + v'_4 . \tag{204}$$

From the projection of the edge connecting P'_1 and P'_3 into the u'w'-plane,

$$u' = u'_3 - \frac{(u'_3 - u'_1)}{(v'_3 - v'_1)}(v'_3 - v') . \tag{205}$$

Using Equations (202) and (205) gives,

$$u'_5 = u'_3 - \frac{(u'_3 - u'_1)}{(v'_3 - v'_1)}(v'_3 - v'_5) . \tag{206}$$

Again, in the u'w'-plane,

$$u' = u'_2 - \frac{(w' - w'_2)}{(w'_4 - w'_2)}(u'_4 - u'_2) , \tag{207}$$

from which

$$u'_6 = u'_2 - \frac{(w'_5 - w'_2)}{(w'_4 - w'_2)}(u'_4 - u'_2) . \tag{208}$$

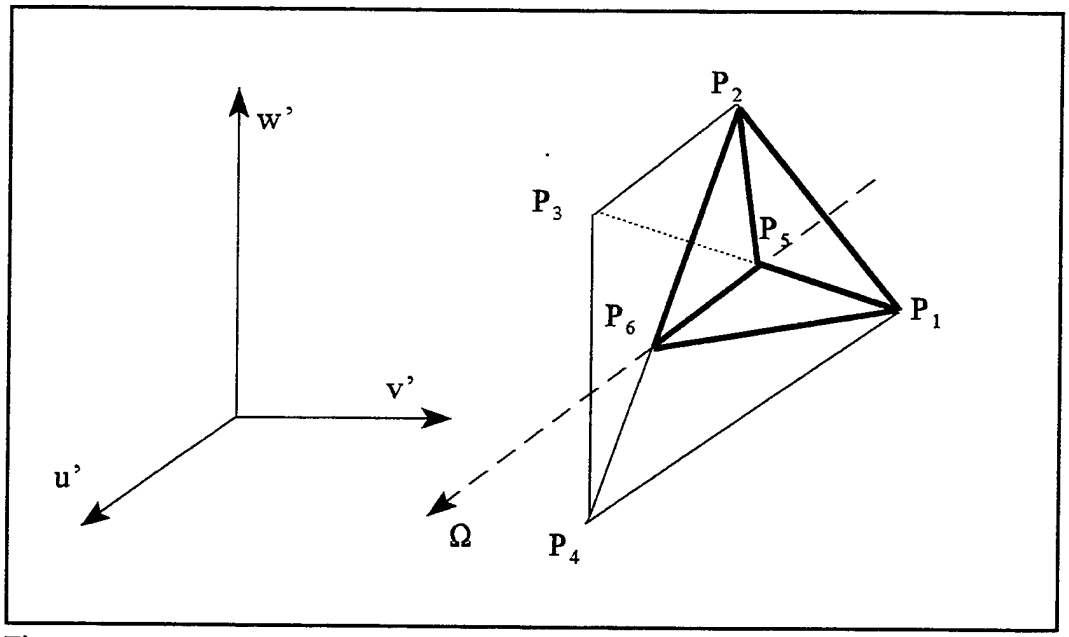

Figure 58. Case 4 Tetrahedron in Sub-Tetrahedron 1 Rotated Coordinate System.

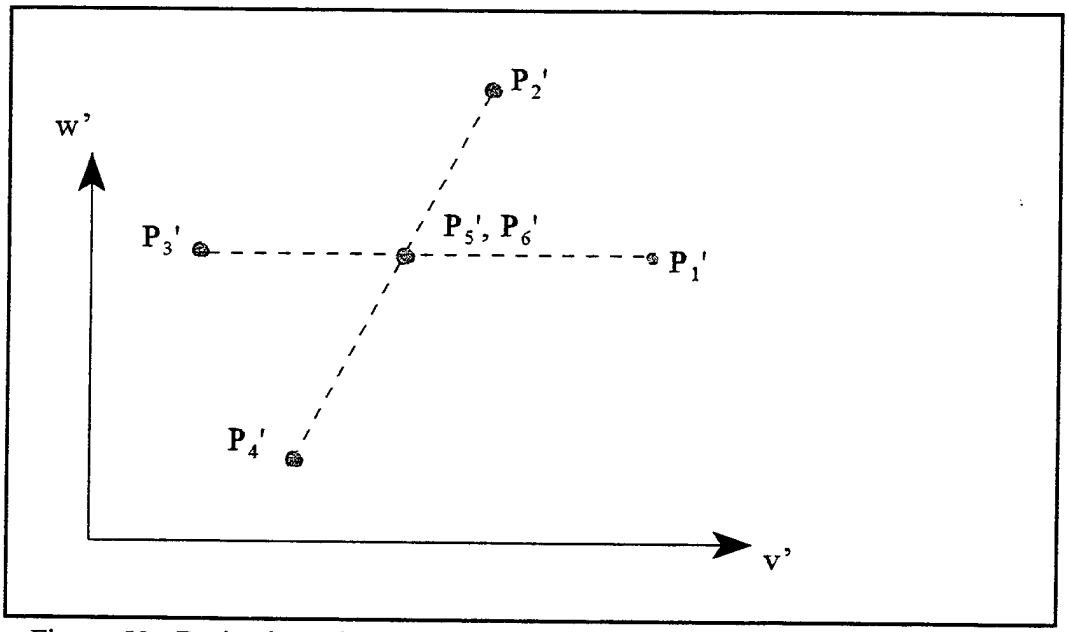

Figure 59. Projection of Case 4, Sub-Tetrahedron 1 Edges into the v'w'-Plane.

We now have all the elements of the sub-tetrahedron coordinate matrix

$$\mathbf{T}'_1 = \begin{pmatrix} u'_5 & u'_6 & u'_1 & u'_2 \\ v'_5 & v'_6 & v'_1 & v'_2 \\ w'_5 & w'_6 & w'_1 & w'_2 \end{pmatrix} \tag{209}$$

and

$$\mathbf{t} = \begin{pmatrix} u'_5 \\ v'_5 \\ w'_5 \end{pmatrix} . \tag{210}$$

At this point all the information is available to transform the tetrahedron into the local uvw system.

Since the position of the u-axis is the same for all four sub-tetrahedra, the values of u'_5 and u'_6 are the same for all sub-tets. Further, since $v'_5=v'_6$ and $w'_5=w'_6$, we only need compute v'_5 and w'_5 and the unit vectors for each of the remaining sub-tets to complete the splitting procedure.

C.2.2 . Sub-Tetrahedron 2 (Configuration 1)

For sub-tetrahedron 2,

$$\hat{e}_u = \hat{\Omega}, \qquad \hat{e}_w = \hat{e}_u \times \frac{\vec{P}_{42}}{|\vec{P}_{42}|}, \qquad \text{and} \qquad \hat{e}_v = \hat{e}_w \times \hat{e}_u . \tag{211}$$

Again, rotate the entire parent tetrahedron into the new u'v'w' coordinate system. From the projection of $\overline{P_1P_3}$ and $\overline{P_2P_4}$ into the v'w' plane, $w'_2 = w'_4 = w'_5 = w'_6$, and

$$v' = \frac{(w' - w'_3)}{(w'_3 - w'_1)}(v'_3 - v'_1) + v'_3, \tag{212}$$

which, at P'_5, gives

$$v'_5 = \frac{(w'_5 - w'_3)}{(w'_3 - w'_1)}(v'_3 - v'_1) + v'_3. \tag{213}$$

C.2.3. Sub-Tetrahedron 3 (Configuration 1)

For sub-tetrahedron 3,

$$\hat{e}_u = \hat{\Omega}, \qquad \hat{e}_w = \hat{e}_u \times \frac{\vec{P}_{13}}{|\vec{P}_{13}|}, \qquad \text{and} \qquad \hat{e}_v = \hat{e}_w \times \hat{e}_u. \tag{214}$$

By symmetry, w'_5 and v'_5 can be found using $w'_1 = w'_3 = w'_5 = w'_6$ and Equation (204).

C.2.4. Sub-Tetrahedron 4 (Configuration 1)

For sub-tetrahedron 4,

$$\hat{e}_u = \hat{\Omega}, \qquad \hat{e}_w = \hat{e}_u \times \frac{\vec{P}_{24}}{|\vec{P}_{24}|}, \qquad \text{and} \qquad \hat{e}_v = \hat{e}_w \times \hat{e}_u. \tag{215}$$

By symmetry, w'_5 and v'_5 can be found using $w'_2 = w'_4 = w'_5 = w'_6$ and Equation (213).

C.2.5 Configuration 2

The equations for Configuration 2 can be derived in a completely analogous manner to the Configuration 1 equations. The resulting equations are identical to the Configuration 1 equations with the exception that u'_5 and u'_6 are computed on Sub-Tetrahedron 1 using

$$u'_5 = u'_1 + \frac{(w'_5 - w'_1)}{(w'_3 - w'_1)}(u'_3 - u'_1), \tag{216}$$

$$\text{and,} \quad u'_6 = u'_2 + \frac{(v'_5 - v'_2)}{(v'_4 - v'_2)}(u'_4 - u'_2). \tag{217}$$

C.3. Case 3 (-3) Tetrahedra Splitting

Case 3 and -3 tetrahedra are characterized by having either three input and one output, or one input and three output, faces. When split along the streaming direction, the Case 3 (-3) tetrahedron subdivides into three Case 1 sub-tetrahedra, as shown in (217). A Case 2 (-2) tetrahedra results when the streaming direction intersects the entering (exiting) face along one of the edges, and a Case 1 tetrahedron results when the intersecting point is coincident with one of the entering (exiting) face apexes.

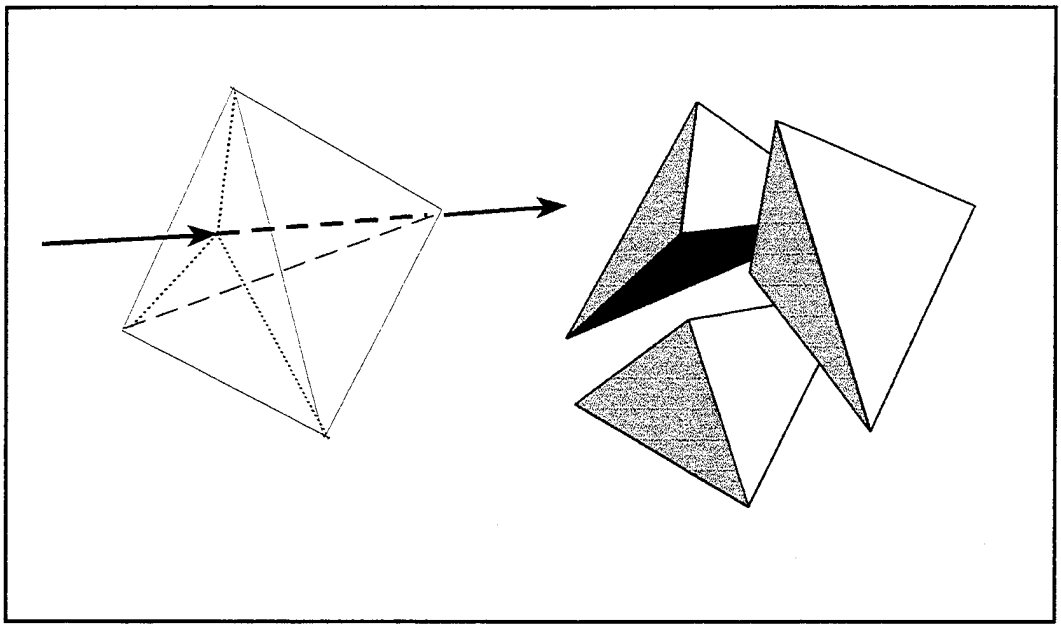

Figure 60. Case 3 Tetrahedron and Resulting Sub-Tetrahedra

Each tetrahedron is defined by four nodes, each having an x, y, and z coordinate in the global xyz coordinate system. There is no a-priori way of knowing how the individual nodes are oriented with respect to one another in space, so the following labeling convention is used. When a tetrahedron is passed to the Case 3 (-3) routines, the nodes defining the entering (exiting) face are labeled P_1, P_2, and P_3, and the remaining node is labeled P_4. P_4 is identified by simply summing the four node numbers defining the tetrahedron, and subtracting the three node numbers defining the entering face. The result is the node number of P_4. Once the intersection point between the streaming direction and the entering (exiting) face is determined, it is labeled P_5. This results in four possible configurations, shown in Figure 61, where the configuration numbers refer to the

orientation of points P_1, P_2, and P_3 about the point P_5 on the outward side of the entering (exiting) face. The orientation is determined using

$$\left(\vec{P}_{12} \times \vec{P}_{13}\right) \cdot \hat{n} > 0, \textit{Configuration } 1$$

$$\left(\vec{P}_{12} \times \vec{P}_{13}\right) \cdot \hat{n} < 0, \textit{Configuration } 2$$

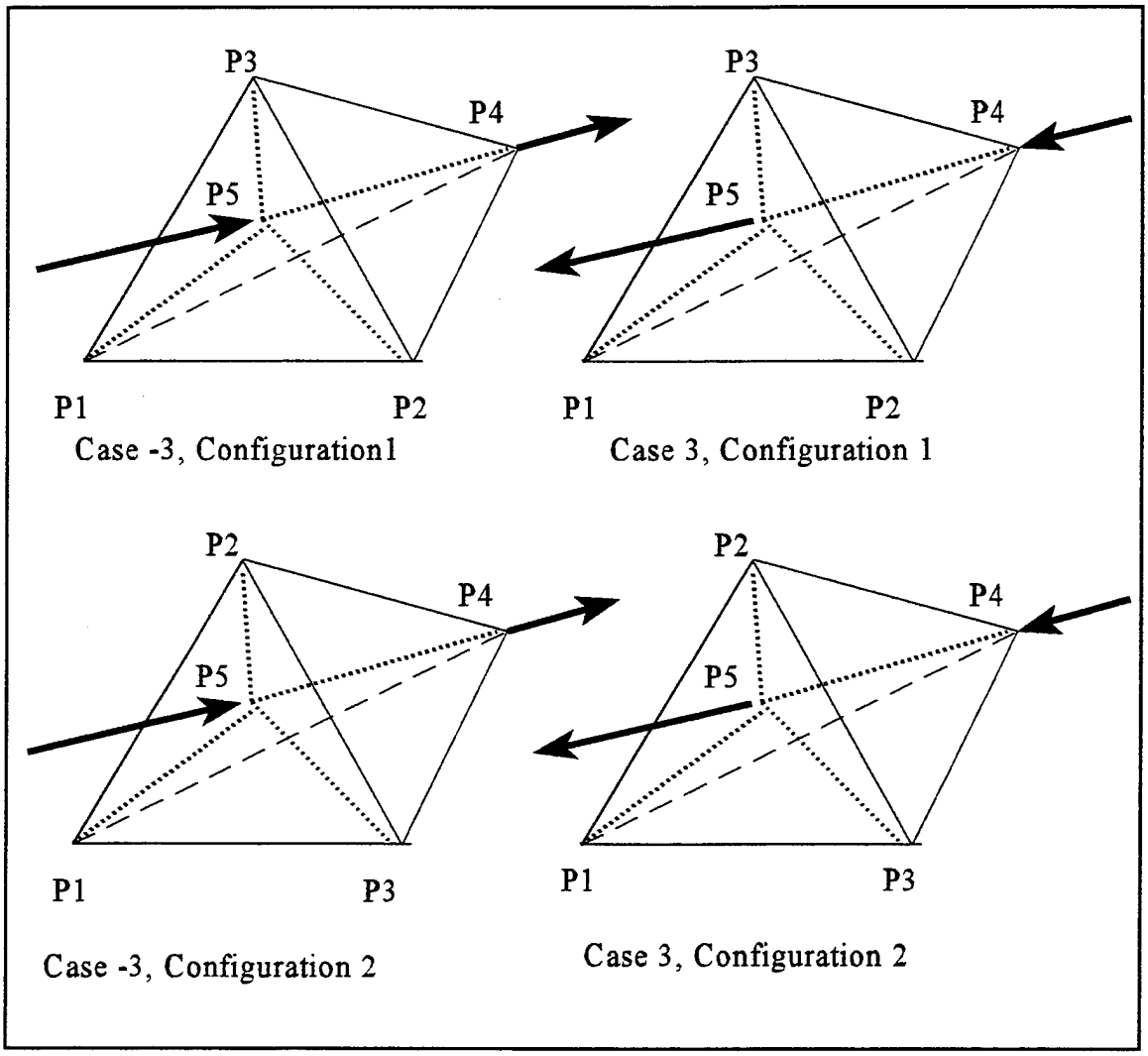

Figure 61. Case 3 Tetrahedra and Possible Point Labeling Configurations.

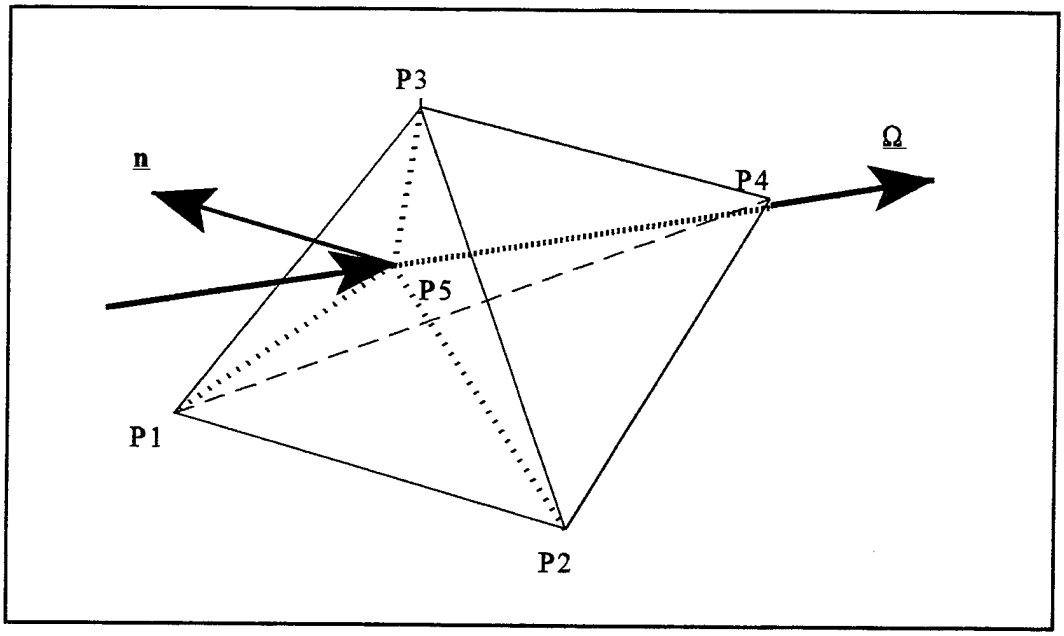

Figure 62. Case -3 Tetrahedron

where $\vec{P}_{12}$ and $\vec{P}_{13}$ are vectors from P_1 to P_2 and from P_1 to P_3 respectively, and $\hat{n}$ is the entering (exiting) face outward normal. Once the specific configuration is known, the orientation of the tetrahedra and the relative position of all the nodes are known, and the sub-tetrahedra and rotation/translation matrices can be generated. In the following discussion, we use Case -3, Configuration 1 as an example. Figure 62 shows a Case -3 tetrahedron. The first step in splitting the tetrahedron along the streaming direction is to determine the coordinates of the point P_5, where the streaming direction intersects the entering face. Mathematically this is equivalent to finding the intersection between a line parallel to Ω passing through point P_4, and the plane defined by points P_1, P_2, and P_3. The equation for line parallel to Ω passing through P_4 is

$$\frac{x_5 - x_4}{\mu} = \frac{y_5 - y_4}{\eta} = \frac{z_5 - z_4}{\xi} . \tag{219}$$

The gives two independent equations in the three unknowns x_5, y_5, and z_5. We also know that P_5 must fall in the plane P_1 P_2 P_3, which means that the vector from P_1 to P_5, $\vec{P}_{15}$, has to be orthogonal to the face normal, $\hat{n}$, where

$$\hat{n} = n_x \hat{e}_x + n_y \hat{e}_y + n_z \hat{e}_z , \tag{220}$$

and the third equation is then

$$(x_5 - x_1)n_x + (y_5 - y_1)n_y + (z_5 - z_1)n_z = 0 .$$

This gives the following system of three equations in the three unknowns x_5, y_5, and z_5,

$$\begin{aligned} \eta(x_5 - x_4) - \mu(y_5 - y_4) &= 0 \\ \xi(y_5 - y_4) - \eta(z_5 - z_4) &= 0 \\ (x_5 - x_1)n_x + (y_5 - y_1)n_y + (z_5 - z_1)n_z &= 0 . \end{aligned} \tag{222}$$

Solving the system in Mathematica and simplifying using vector notation gives

$$x_5 = \mu \frac{\hat{n} \cdot \vec{P}_{41}}{\hat{n} \cdot \hat{\Omega}} + x_4$$

$$y_5 = \eta \frac{\hat{n} \cdot \vec{P}_{41}}{\hat{n} \cdot \hat{\Omega}} + y_4 \qquad (223)$$

$$z_5 = \xi \frac{\hat{n} \cdot \vec{P}_{41}}{\hat{n} \cdot \hat{\Omega}} + z_4 \,.$$

In Equations (223), μ, η, ξ, x_4, y_4, z_4, and $\hat{n} \cdot \hat{\Omega}$, are all known. The only quantity that has to be computed is $\hat{n} \cdot \vec{P}_{41}$, which need only be computed once to get the three coordinates. This same system of equations is used for both Case 3 and Case -3, and for both configurations. Given the coordinates of P_5, all the necessary information is available to generate the transformation matrices and transform the sub-tetrahedra into the local coordinate system.

Equations (223) can also be obtained by geometric arguments. Equations (223) are equivalent to

$$\vec{P}_{45} = \vec{P}_5 - \vec{P}_4 = \hat{\Omega} \left(\frac{\hat{n} \cdot \vec{P}_{41}}{\hat{n} \cdot \hat{\Omega}} \right) \qquad (224)$$

Examining Figure 63 we see that the altitude is $b = \vec{P}_{41} \cdot \hat{n}$ and that the edge is $c = b / \cos\theta$ where $\cos\theta = \hat{n} \cdot \hat{\Omega}$, thus

$$c = \frac{\vec{P}_{41} \cdot \hat{n}}{\hat{n} \cdot \hat{\Omega}} \,. \qquad (225)$$

Equation (224) follows from the fact that $\vec{P}_{45} = c\,\hat{\Omega}$.

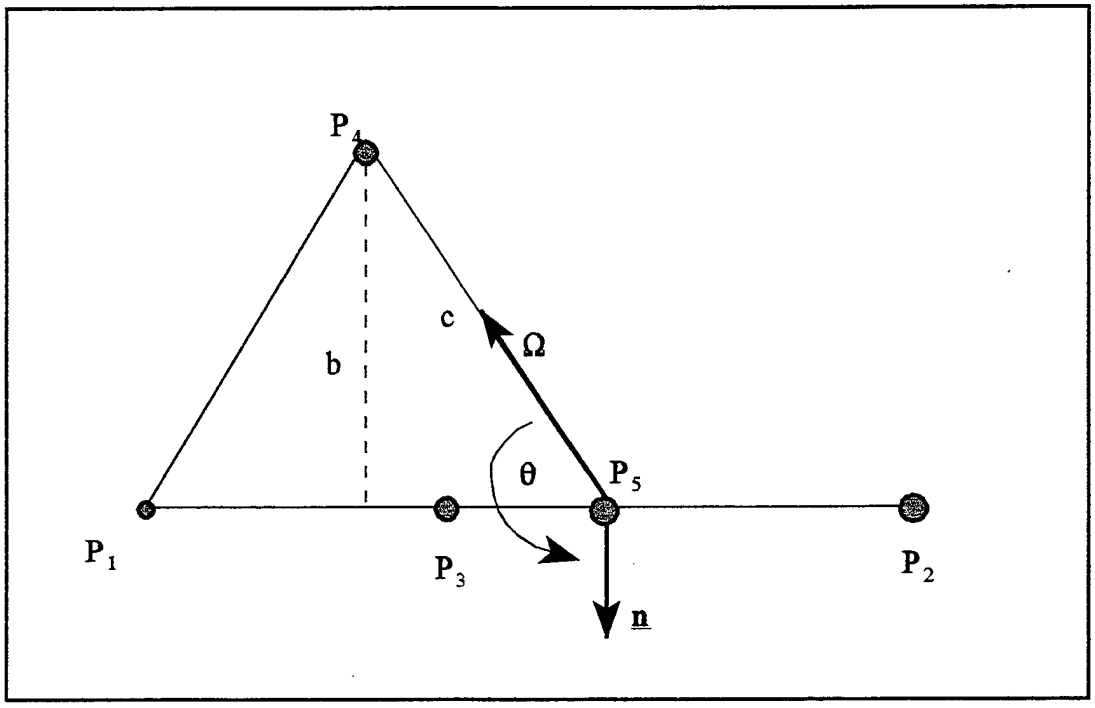

Figure 63. Side view of Case 3 Tetrahedron.

C.4. Case 2 (-2) Tetrahedra Splitting

Case 2 (-2) tetrahedra are characterized by having either two input and one output, or one input and two output faces. When split along the streaming direction, a Case 2 (-2) tetrahedron produces two, Case 1 tetrahedra, as shown in Figure 64. The Case 2 tetrahedron is a degenerate case of the Case 3 tetrahedron. The labeling convention for the Case 2 tetrahedron is the same as that used for Case 3, with the exception that the point P_5 is restricted to lie along $\overline{P_1P_2}$. As with Case 3, this results in two possible configurations each for Case 2 and Case -2. The splitting is handled exactly as with the

Case 3 tetrahedron, the coordinates of point P_5 are found using Equation (223). Given these coordinates, all the necessary information is available to generate the transformation matrices and transform the sub-tetrahedra into the local coordinate system.

C.5. Case 1 Sub-tetrahedra

Case 1 tetrahedra are characterized by having only one input and one output face. No splitting is necessary since the tetrahedron already has the desired special case characteristics.

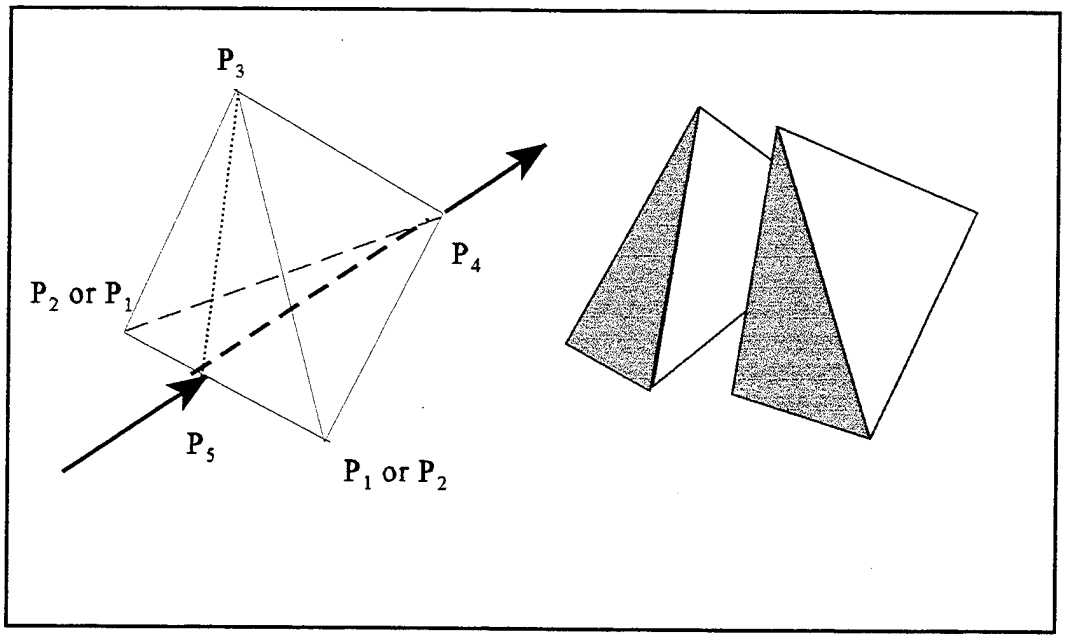

Figure 64. Case 2 Tetrahedron and Resulting Sub-Tetrahedra.

Appendix D. Conservation Equations

Solved for the cell flux moments, the balance equations are

$$\psi^A = \frac{u_1 S^A + 3\psi_A^{in}}{\epsilon} - \frac{3\psi_A^{out}}{\epsilon},$$

$$\psi^w = \frac{u_1 S^w + 3\psi_w^{in}}{\epsilon} - \frac{3\psi_w^{out}}{\epsilon},$$

$$\psi^v = \frac{u_1 S^v + 3\psi_v^{in}}{\epsilon} - \frac{3\psi_v^{out}}{\epsilon}, \quad \text{and} \tag{226}$$

$$\psi^u = \frac{u_1 S^A + \psi^A + 3\psi_u^{in}}{\epsilon} - \frac{3\psi_u^{out}}{\epsilon}.$$

In this form, the balance equations provide a much more computationally efficient means of computing the subcell flux moments since all quantities on the right hand side of Equations (226) are known. For example, to compute the subcell flux moments for the EC method using Equations (89) and (91) the code must compute three distinct three-argument moment functions and four, four-argument moment functions. The same calculation using the balance equations is essentially free. Note that for LC, CLC, and SC, all of the moments functions needed to compute the subcell flux moments must be computed to get the source and entering flux moments, so there is no efficiency gain in using the balance equations for these methods.

There is a drawback to using Equations (226). As the cell optical thickness approaches zero ($\epsilon \to 0$) they become ill-conditioned, and loss of significant digits can

occur. Note that the equations are all in the form a-b. We can estimate the precision lost by examining the quantity

$$n = \log\left(\frac{|a-b|}{|a+b|} \right). \tag{227}$$

If n is less than zero, then loss of digits due to subtraction of nearly equal quantities has occurred resulting in a loss of approximately n digits. We can use Equation (227) to limit the precision lost in computing the cell moments using the balance equations by requiring

$$\psi^j < (a+b)10^m \tag{228}$$

where j=A, u, v, or w, and m is the maximum number of digits allowed to be lost. For all problems computed in Chapter 5, the moments functions were computed to twelve digits, and the source and entering flux distribution coefficients were computed to ten digits, therefore it is not reasonable to demand 16 digits of accuracy in the cell flux moments computation. At most, the cell moments need only be as accurate as the moment function computation tolerance. Further, the final results are less sensitive to the number of accurate digits in the cell first moments than in the number of accurate digits in the average moments, hence the first moments do not have to be computed to as much accuracy as the zeroth moments.

The EC quadrature routine using the balance equations works as follows:

Compute the exiting flux moments

Compute the source moments using the distribution coefficients

Compute the entering flux moments using the distribution coefficients

Compute the cell flux moments using the balance equations

If (lost > 4 digits for zeroth or > 6 digits for first moments) then

Recompute cell moments using Equations (89) and (91)

Endif.

Appendix E. Mesh Generation

E.1 IDEAS

Tetrahedral meshes were generated using the IDEAS (Lawry, 1991) code. IDEAS is a finite element code with the capability to generate three-dimensional meshes using a variety of cell geometries. *The IDEAS Student Guide* (Lawry, 1991) and the set of IDEAS manuals contain information on generating meshes. Once generated, the data are written to a file in NASTRAN format. The NASTRAN data file is more streamlined than the IDEAS ASCII output. This both reduces the size of the files that have to be manipulated and simplifies the post processing. A sample NASTRAN output for the six tetrahedron cube problem is shown in Section E.2, and the TetSN input file generated from the NASTRAN file is shown in Section E.3.

E.2 NASTRAN Output File

```
BEGIN BULK
CORD2S*                 2               0      0.000000000      0.000000000+1A     2
*1A     2      0.000000000      0.000000000      0.000000000      100.0000000+1B     2
*1B     2      100.0000000      0.000000000      0.000000000               +1C     2
*1C     2
CORD2C*                 1               0      0.000000000      0.000000000+1A     1
*1A     1      0.000000000      0.000000000      0.000000000      100.0000000+1B     1
*1B     1      100.0000000      0.000000000      0.000000000               +1C     1
*1C     1
GRID*                   1               0     -0.500000000     -0.500000000+GA     1
*GA     1      0.500000000               0
GRID*                   2               0     -0.500000000      0.500000000+GA     2
*GA     2      0.500000000               0
GRID*                   3               0      0.500000000     -0.500000000+GA     3
*GA     3      0.500000000               0
GRID*                   4               0      0.500000000      0.500000000+GA     4
*GA     4      0.500000000               0
GRID*                   5               0     -0.500000000     -0.500000000+GA     5
*GA     5     -0.500000000               0
GRID*                   6               0     -0.500000000      0.500000000+GA     6
*GA     6     -0.500000000               0
GRID*                   7               0      0.500000000     -0.500000000+GA     7
*GA     7     -0.500000000               0
GRID*                   8               0      0.500000000      0.500000000+GA     8
*GA     8     -0.500000000               0
CTETRA          1       2       4       7       3       5
CTETRA          2       2       4       3       1       5
CTETRA          3       2       4       1       2       5
CTETRA          4       2       4       2       6       5
CTETRA          5       2       4       6       8       5
CTETRA          6       2       4       8       7       5
MAT1*                   1     2.068000E+9                      0.289999992+MA     1
*MA     1      0.007820000      0.000011700                               +MB     1
*MB     1      1.500000E+7      1.500000E+7      680000.0000              +MC     1
*MC     1
PSOLID*                 2               1               0               +PA     2
*PA     2
PARAM   AUTOSPC YES
PARAM   POST    -2
ENDDATA
```

E.3 Sample TetSN Input File

```
6 tetrahedron cube
# NODES
8
X,Y,Z COORDINATES OF EACH NODE
-0.50000000000000000 -0.50000000000000000 0.50000000000000000
-0.50000000000000000 0.50000000000000000 0.50000000000000000
0.50000000000000000 -0.50000000000000000 0.50000000000000000
0.50000000000000000 0.50000000000000000 0.50000000000000000
-0.50000000000000000 -0.50000000000000000 -0.50000000000000000
-0.50000000000000000 0.50000000000000000 -0.50000000000000000
0.50000000000000000 -0.50000000000000000 -0.50000000000000000
0.50000000000000000 0.50000000000000000 -0.50000000000000000
NUMBER OF TETRAHEDRA
6
NODES DEFINING EACH TETRAHEDRA
4 7 3 5
4 3 1 5
4 1 2 5
4 2 6 5
4 6 8 5
4 8 7 5
MATERIAL NUMBER OF EACH TETRAHEDRA
1
1
1
1
1
1
NUMBER OF MATERIALS
1
NUMBER OF ENERGY GROUPS
1
SIGMA-TOTAL, SOURCE, SIGMA-SCATTER
1.0000000000000 1.00000000000000000
5.0D-1
BOUNDARY FACE DATA
0
```

Bibliography

Abu-Shumays, I.K, . "Compatible Product Angular Quadrature for Neutron Transport in X-Y Geometry", Nuclear Science and Engineering, 64: 299, (1977).

Adams, Marvin L., "A New Transport Discretization Scheme for Arbitrary Spatial Meshes in XY Geometry", Intl. Topl. Mtg. Advances in Mathematics, Computations, and Reactor Physics, Pittsburgh, PA., April 28 - May 2, 1991, American Nuclear Society, (1991).

Alcouffe, Raymond E., "THREEDANT: A Code to Perform Three-Dimensional, Neutral Particle Transport Calculations", Proc. Topl. Mtg. International Conference on Mathematics and Reactor Physics, and Environmental Analyses, Portland OR, April 30- May 4, American Nuclear Society, 1, 461, (1995).

Azmy, Y. Y., "The Weighted Diamond Difference Form of Nodal Transport Methods", Nuclear Science and Engineering, 98: 29-40, (1988).

Barbucci, P. and F. Di Pasquantonio, "Exponential Supplementary Equations for SN Methods: The One-Dimensional Case", Nuclear Science and Engineering, 63: 179-187, (1977).

Briesmeister, Judith F., MCNP - A General Monte Carlo Code for Neutron and Photon Transport, Los Alamos National Laboratory, Los Alamos NM, 1991.

Carlson, B. G. and G. I. Bell, "Solution of the Transport Equation by the SN Method", Proc. U. N. Intl. Conference on the Peaceful Uses of Atomic Energy, Geneva, 2, 2386 (1958)

Carlson, B. G. and K. D. Lathrop, "Transport Theory - The Method of Discrete Ordinates", Computing Methods in Reactor Physics, Greenspan, Kelber and Okrent Editors, Gordon and Breech, New York, (1968).

Castrianni, C. L. and M. L. Adams, "A Nonlinear Corner-Balance Spatial Discretization for Transport on Arbitrary Grids", Proc. Topl. Mtg. International Conference on Mathematics and Reactor Physics, and Environmental Analyses, Portland OR, April 30- May 4, American Nuclear Society, 2, 916, (1995).

Chandrasekhar, S. Radiative Transport, Dover, New York, (1960).

Davison, B., Neutron Transport Theory, London: Oxford University Press, 1958.

Duderstadt, J. J., and L. J. Hamilton, Nuclear Reactor Analysis, New York: John Wiley and Sons, 1976.

Groves, R. E. and R. E. Pevey, "A Characteristic Based Multiple Balance Approach for S_N on Arbitrary Polygonal Meshes", Proc. Topl. Mtg. International Conference on Mathematics and Reactor Physics, and Environmental Analyses, Portland OR, April 30- May 4, American Nuclear Society, 2, 928, (1995).

Larsen, E. W. and R. E. Alcouffe, "The Linear Characteristic Method for Spatially Discretizing the Discrete-Ordinates Equations in x, y Geometry", Proc. Topl. Mtg. Advances in Mathematical Methods for the Solution of Engineering Problems, Munich, FRG, April 27-29, 1981, American Nuclear Society, 1, 99, (1981).

Larsen, E. W. and W. F. Miller, "Convergence Rates of Spatial Difference Equations for the Discrete-Ordinates Neutron Transport Equations in Slab Geometry", Nuclear Science and Engineering, 45: 76-83 (1980).

Lathrop, K. D., "Spatial Differencing of the Transport Equation: Positivity vs Accuracy", J. Comp. Phys. 4: 475-498 (1969).

Lawrence, R. D., "Progress in Nodal Methods for the Solution of the Neutron Diffusion and Transport Equations", Progress in Nuclear Energy, 17: 271-301, (1986).

Lawry, Mark H., I-DEAS Student Guide, Milford Ohio: Structural Dynamics Research Corporation, 1991.

Lewis, E. E., and Miller, W. F. Jr., Computational Methods of Neutron Transport, La Grange Park, Illinois: American Nuclear Society, 1993.

Mathews, K. A., Discrete Elements Neutral Particle Transport, Ph.D. Dissertation, AFIT/DS/83-5. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH. October, 1983.

Mathews, K. , Sjonden, G., and Minor, B. ,"Exponential Characteristic Spatial Quadratures for Discrete Ordinates Radiation Transport in Slab Geometry", Nuclear Science and Engineering, 118: 24-37 (1994).

Mathews, K.A. and C. R. Brennan, "Exponential Characteristic Spatial Quadrature for Discrete Ordinates Radiation Transport on an Unstructured Grid of Triangular Cells", Proc. Topl. Mtg. International Conference on Mathematics and Reactor Physics, and Environmental Analyses, Portland OR, April 30- May 4, 1995, American Nuclear Society.

Mathews, Kirk A., LCDR, USN, "Adaptive Characteristic Spatial Quadratures for Discrete Ordinates Neutral Particle Transport - The Slab Geometry Case", Transport Theory and Statistical Physics, 19: 419-458, (1990).

Mathews, K. A. and B. M. Minor, "Step Adaptive Characteristic Spatial Quadrature for Discrete Ordinates Neutral Particle Transport in Two-Dimensional Cartesian Coordinates", Intl. Topl. Mtg. Advances in Mathematics, Computations, and Reactor Physics, Pittsburgh, PA., April 28 - May 2, 1991, American Nuclear Society, (1991).

Mathews, K. A. and B. M. Minor, "Adaptive Characteristic Spatial Quadratures for Discrete Ordinates Neutral Particle Transport - The Rectangular Cell Case", Transport Theory and Statistical Physics, 22: 655-686, (1993).

Mathews, K.A., Minor, B. M. , and C. R. Brennan, "Exponential Moments Functions-Applications and Numerical Evaluation", American Nuclear Society 1995 Annual Meeting, June 25-29, 1995, Philadelphia, PA.

Miller, D.J., K. A. Mathews and C. R. Brennan, "Split Cell Discrete Ordinates Transport on and Unstructured Grid of Triangular Cells", Accepted for Publication in Transport Theory and Statistical Physics, January, 1996.

Miller, Dennis J., Linear Characteristic Spatial Quadrature for Discrete Ordinates Neutral Particle Transport on Arbitrary Triangles, Ph.D. Dissertation AFIT/DS/ENP/93-02. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH. June 1993.

Minor, Bryan M. ,Exponential Characteristic Spatial Quadrature for Discrete Ordinates Neutral Particle Transport in Two-Dimensional Cartesian Coordinates, Ph.D. Dissertation AFIT/DS/ENP/93-7. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH. September 1993.

Petrovic, B. G. and A. Haghighat, "Boundary Conditions Induced Oscillations in the S_N Solutions of the Two-Dimensional Neutron Transport Equation", Proc. Topl. Mtg. International Conference on Mathematics and Reactor Physics, and Environmental Analyses, Portland OR, April 30- May 4, 1995, American Nuclear Society.

Press, W. H., T. W. Vettering, S. A. Teukolsky, and B. P. Flannery, Numerical Recipes in Fortran (Second Edition), Cambridge University Press, 1992.

Rhoades, Wayne A. and Yousry Y. Azmy, "Three Dimensional SN Calculations with the Oak Ridge TORT Code", Proc. Topl. Mtg. International Conference on Mathematics and Reactor Physics, and Environmental Analyses, Portland OR, April 30- May 4, American Nuclear Society, 1, 480, (1995).

Sjonden, G. E., Exponential Characteristic Spatial Quadratures for Discrete Ordinates Neutral Particle Transport in Slab Geometry, M. S. Thesis, AFIT/GNE/ENP/92M-10. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH. March 1992.

Walters, Wallace F., "Augmented Weighted-Diamond Form of the Linear Nodal Scheme for Cartesian Coordinate Systems", Nuclear Science and Engineering, 92: 192-196 (1986).

Walters, Wallace F., "The Relationship Between Finite Elements and Nodal Methods in Transport Theory", Progress in Nuclear Energy, 18: 21-26 (1986).

Walters, Wallace F. LANL, Private Communication, 16 June 1994.

Walters, W. F. and R. D. O'Dell, "Nodal Methods for Discrete Ordinates Transport Problems in (XY) Geometry", Proc. Topl. Mtg. Advances in Mathematical Methods for the Solution of Engineering Problems, Munich, FRG, April 27-29, 1981, American Nuclear Society, 1, 115, (1981).

Walters, W. F., and T. A. Wareing, "A Non-Linear Positive Method for Solving the Transport Equation on Coarse Meshes," Proceedings Eighth International Conference on Radiation Shielding, Vol 1, Arlington, Texas, 1994.

Walters, W. F. , T. A. Wareing, and D. R. Marr, "The Non-Linear Characteristic Scheme for X-Y Geometry Transport Problems", Proceedings International Conference on Mathematics and Computations, Reactor Physics and Environmental Analyses, Portland, Oregon, April 30-May 4, 1995.

Wareing, T. A., W. F. Walters and J. E. Morel, "A Diffusion Accelerated Solution Method for the Nonlinear Characteristic Scheme", Proc. International Conference on Mathematics, Reactor Physics, and Environmental Analyses, Portland, Oregon, April 30 - May 4, 1995, 2, 1335, (1995).

Wickham-Jones, Tom, Mathematica Graphics; Techniques and Applications, New York: Springer-Verlag, 1994.

Wolfram Research Inc., Mathematica Version 2.2, Champaign Ill: Wolfram Research Inc., 1992.

Vita

Captain Charles Brennan was born on February 8, 1961 in Philadelphia, Pennsylvania. He enlisted in the Air Force in September of 1981 and was initially assigned to Moody AFB, Georgia, as an inertial navigation technician on the F4 aircraft. He was accepted into the Airman Education and Commissioning Program in 1985 and was sent to the University of Florida where he graduated with a Bachelor's degree in Nuclear Engineering in August of 1987. He received his commission from Officer Training School in December of 1987 and was assigned to the Air Force Weapons Laboratory, Kirtland AFB, New Mexico, where he served as a test director at the electromagnetic pulse test facilities. He was selected to attend the Air Force Institute of Technology in August of 1991, and was awarded a Master's degree in Nuclear Engineering in March of 1993, and a Ph.D. in June of 1996.

Permanent Address: 9995 Arnold Rd.
Jacksonville Florida, 32246

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1996	Doctoral Dissertation

4. TITLE AND SUBTITLE
CHARACTERISTIC SPATIAL QUADRATURES FOR DISCRETE ORDINATES NEUTRAL PARTICLE TRANSPORT ON ARBITRARY TETRAHEDRAL MESHES

5. FUNDING NUMBERS

6. AUTHOR(S)

Charles R. Brennan, Captain, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
2950 P Street
WPAFB OH, 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/DS/ENP/96-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Headquarters, Defense Nuclear Agency
6801 Telegraph Road
Attn: Clifton McFarland
Director, Weapons Effects
Alexandria, VA 22310-3398

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Characteristic spatial quadratures for discrete ordinates calculations on meshes of arbitrary tetrahedra are derived and tested, including the step (SC), linear (LC), and exponential (EC) characteristic quadratures and variants that assume constant distributions on cell faces. Tetrahedral meshes accurately model curved surfaces with few cells. A split cell approach subdivides tetrahedra along the streaming direction, reducing the transport to one dimension. Assumed forms of the cell source and entering flux distributions have sufficient parameters to match the zeroth and first spatial moments. These parameters are determined by analytically inverting a linear system (LC), or by numerical inversion using Newton's method (EC). Efficient algorithms for the two- and three-dimensional rootsolves are derived. The constant face methods proved unacceptable in empirical testing. Both LC and EC exhibited third order convergence. LC provided accurate results on cells with optical thickness on the order of one mean free path while EC was accurate with fewer, thicker cells. LC can produce negative fluxes; EC is strictly positive. Although more costly per cell, EC is robust and can be more efficient on coarse meshes.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Boltzmann Equation, Neutron Transport Theory, Numerical Methods, Unstructured Grids, Radiation Transport, Radiation Shielding	228
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED